A Novel Framework of Board-Level Failure Localization in Optical Transport Networks

by

Yan Jiao

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2024

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner:	Xiaohong Jiang Professor, School of Systems Information Science, Future University Hakodate
Supervisor(s):	Pin-Han Ho Professor, Dept. of Electrical and Computer Engineering, University of Waterloo
Internal Member:	Otman Basir Professor, Dept. of Electrical and Computer Engineering, University of Waterloo
Internal Member:	Zhou Wang Professor, Dept. of Electrical and Computer Engineering, University of Waterloo
Internal-External Member:	Stephen Vavasis Professor, Dept. of Combinatorics and Optimization, University of Waterloo

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

Chapter 2, 3, and 4 incorporate the work from four papers. As the first author of these papers, I was responsible for proposing the framework, writing codes regarding machine learning models and integer linear programing solver, conducting main experiments, and finalizing the manuscripts. My co-author, Xiangzhu Lu, contributed to writing codes for the simulator. Kairan Liang contributed to running the machine learning models. Yuren You was the OTN equipment provider. Prof. Pin-Han Ho, Prof. János Tapolcai, Prof. Limei Peng, and Prof. Bingbing Li provided guidance and feedback on my draft manuscripts. These papers are listed below:

- Y. Jiao, P. -H. Ho, X. Lu, J. Tapolcai, and L. Peng, "A Novel Framework for Optical Layer Device Board Failure Localization in Optical Transport Network," in *IEEE Transactions on Network and Service Management*, accepted.
- Y. Jiao, P. -H. Ho, X. Lu, J. Tapolcai, and L. Peng, "A Novel Framework of Failure Localization in Optical Transport Network," in *IEEE Communications Magazine*, vol. 61, no. 12, pp. 142–147, 2023.
- Y. Jiao, P. -H. Ho, X. Lu, K. Liang, Y. You, J. Tapolcai, B. Li, and L. Peng, "On Real-time Failure Localization via Instance Correlation in Optical Transport Networks," in *2023 IFIP Networking Conference (IFIP Networking)*, Barcelona, Spain, pp. 1–9, 2023.
- Y. Jiao, P. -H. Ho, X. Lu, J. Tapolcai, and L. Peng, "On Board-level Failure Localization in Optical Transport Networks Using Graph Neural Network", accepted by *2024 International Conference on the Design of Reliable Communication Networks*.

Also, I have co-authored several papers on OTN simulator development, as given by:

- Z. Li, P. -H. Ho, Y. Jiao, B. Li and Y. You, "Design of an OTN-based Failure/Alarm Propagation Simulator", in *2022 International Conference on Networking and Network Applications (NaNA)*, Urumqi, China, 2022, pp. 1–5.
- X. Lu, Y. Jiao, and P. -H. Ho, "Architectural Design and Test Case Analysis of a Simulator for Failure Localization", in *Journal of Networking and Network Applications*, in review.

Abstract

Optical transport networks (OTNs) serve as a pivotal role in Internet backbones thanks to their support for multi-tenant and multi-service environments with high reliability and low cost. A failure event may affect one or multiple boards in OTN that ignite a vast number of alarms, which significantly boosts the complexity of failure localization and alarm analysis. Accordingly, there is an urgent need for a systematic framework that harnesses the known network state and received alarms to achieve effective failure localization.

Alarm correlation has been considered as a representative approach to identifying the dependencies among alarms, aiming at eliminating as many descendent alarms as possible, thereby fulfilling failure localization with much decreased complexity. Nevertheless, existing methods of alarm correlation are subject to the following issues. Firstly, they ignore the fact that alarm propagation mostly takes place along certain connections and that the network topology and traffic distribution may solidly underpin the required alarm correlation process. Secondly, they necessitate heuristically setting initial parameters but lack a general rule that adjusts their values according to various network characters. Lastly, they are deficient in generality to versatile network environments, where the obtained result grounded in a specific network state may not be migrated to another.

Enlightened by its significance and stringent requirements, this thesis proposes a novel framework of board-level failure localization in OTN, called Failure-Alarm Correlation Tree based Failure Localization (FACT-FL). It aims to construct one or multiple FACTs that achieve both failure localization and alarm correlation, where each FACT takes a failed board and its associated alarms as the tree root and leaves, respectively. We have designed three methodologies to obtain viable FACTs. A scheme named FACT-FL-Heuristic is firstly attempted via a learned binary classifier that intelligently captures the historical correlations in the form of board $\rightarrow$ alarm and alarm $\rightarrow$ alarm, followed by heuristically creating the feasible FACT(s). To further improve FACT-FL-Heuristic's performance, a method termed FACT-FL-Chain treats each FACT as a suite of correlation chains with different order values and generates viable FACT(s) by elegantly solving an integer linear programing (ILP) problem. Moreover, to reduce the computational complexity incurred by enumerating all chain candidates with FACT-FL-Chain, an approach dubbed FACT-FL-GNN leverages graph neural network (GNN) for evaluating the edge weights of potential FACT(s), which facilitates formulating an alternative simplified ILP to yield the most likely FACT(s). The above three methods share the same functional blocks including feature extraction, binary classifier training, and FACT formation, while each method realizes each functional block with different strategies. Extensive case studies are conducted to unveil the proposed methods' advantage over their counterparts in terms of the metrics

assessing the recognized failed boards/root alarms. We also explore their performance in volatile environmental variations such as diverse failure scenarios, network topologies, traffic distributions, and noise alarms.

Acknowledgments

I would like to express my heartfelt gratitude to my supervisor, Prof. Pin-Han Ho, for his careful guidance and financial support over the past five years, and also his invaluable advice for my future career.

I would like to thank all my paper co-authors for their efforts in conducting experiments and paper revision.

I would like to thank my family for their support and encouragement on pursuing my PhD degree.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Telecommunication networks, notably the Internet optical backbones, have gone through tremendous expansion in the past decades not only in their capacity and geographical coverage but also in their heterogeneous nature where a multi-service and multi-tenant environment is supported. Optical transport network (OTN) is a standard control and management protocol introduced by ITU-T that serves as the basis of facilitating such expansion by multiplexing various service flows over individual optical flows [29]. The OTN control plane provides a strong capability in handling end-to-end lightpaths, where it offers abundant maintenance and management signals in the electrical and optical layer overheads. It introduces a suite of rigid alarming mechanisms at each device board in response to any failure event detected by the board sensor. For example, a transponder board triggers an alarm reported to the network management system (NMS) when any irregularity affecting the quality of the received lightpath is identified in the electrical/optical domain. Another example is that the failure of a fiber segment would be recognized by its downstream fiber interface unit (FIU) board which in turn reports an alarm to the NMS.

In general, a failure event may hit one or a set of boards, which probably causes quality of service (QoS) degradation or even service outage resulting in the loss of a significant amount of data. [50] Therefore, failure localization is one of the most important issues in the OTN operation, where the network operator aims to accurately identify the broken network elements and restore them sooner.

However, as the current Internet distributes over thousands of kilometers with millions of network entities, failure localization becomes intractable because a mass of failures hit

the interconnected nodes in the network every day. A failure event could disrupt numerous optical signals, which would be sensed by many other boards that are traversed by these optical signals. An alarm could be raised and/or reported at a board not only owing to an identified failure event but also in response to a notification alarm issued by another remote board. Then these alarms are sent to the centralized NMS which generally receives approximately 1 million alarms every day that overwhelm the network operators. [69] Traditionally, the diagnosis of network failure counts on experienced network operators who are required to manually figure out the relations among alarms and timely identify the root alarm(s) from the alarm storm for locating the network failure(s). [45] For a large-scale network, this poses a great challenge to failure localization procedure and increases the maintenance cost as well. This situation becomes even worse in the network domains of large geographical coverage and huge number of device boards such as nowadays Internet OTN backbones.

1.2 Motivation and Challenges

To reduce the heavy workload of network operators, the key component of failure localization called *alarm correlation* is introduced, which aims to characterize the dependency/causality of collected alarms. It can remarkably decrease the number of alarms by discarding those dependent ones, which facilitates screening out the root cause one(s) for localizing the failure(s). Nevertheless, the above alarm correlation procedure may bring about high computational complexity in the context of an enormous number of alarms propagating in the network. This especially becomes troublesome in a heterogeneous network like the OTN, where the alarm attributes vary with layers and network domains, even if these alarms are emitted from identical hardware devices/boards.

Alarm correlation has been considered as an effective approach to identifying the dependencies among alarms, aiming to remove as many dependent alarms from the failure localization process as possible such that the failure event(s) can be inferred/pinpointed with significantly reduced complexity. By leveraging various alarm correlation techniques, numerous methods have been reported in the literature, most of which focused on the relations among alarms [59, 54, 32, 33, 51, 1, 8, 81, 19, 53, 42, 69, 84, 2, 12, 35, 3, 66, 82], between possible faulty network devices and alarms [50, 75, 77, 45, 46, 47, 76, 85, 36, 74], or between device performance data and device states [20, 70, 71, 78, 79, 34] in a static network environment. However, they are subject to a couple of issues.

- They don't consider the fact that the alarm propagation mostly occurs along certain

connections and that the information on network topology and instantaneous traffic distribution may solidly facilitate the desired alarm correlation process.

- They are required to heuristically set the initial parameters but lack a general rule to determine their values that conform to various network characteristics.
- They lack sufficient generality to various network environments, where the obtained result according to a specific network at a given moment may not be migratable to another.

The aforementioned issues, along with many others, are specific to the OTN optical layer but have not been sufficiently addressed by existing research works, leading to lower precision and/or much more computation in failure localization via identifying correlated alarms. Therefore, a novel failure localization framework via alarm correlation is expected to have the following characteristics:

- Superior performance in localizing faulty board(s)/root alarm(s).
- Sufficient generality to various network environments and adaptability to the changing network topology and traffic distribution.
- High scalability and computational efficiency.

1.3 Related Works

Existing studies on alarm correlation generally employ approaches such as expert systems [59, 54, 32], graphs [81, 8], pattern mining [19, 42, 69, 84], and machine learning [68, 66, 82, 50, 75, 77, 46, 47, 45, 22, 35, 85, 76, 3, 36].

Expert systems represent some of the earliest techniques that automatically transfer the knowledge of correlating alarms from human experts into an exhaustive rule database [59]. However, they come with the significant drawback of high costs while updating the knowledge base, and they struggle to adapt to the network environmental changes. In [54], an expert system called DANTES integrates structural and heuristic knowledge into rules and applies them to network maintenance and problem diagnosis. Afterward, the authors in [32] implement the expert system, namely IMPACT, and employ it for intelligent alarm filtering, alarm generalization, and fault diagnosis.

Graph-based methods model dependencies between alarm types using directed graphs. However, these methods mostly concern temporal relationships of alarms, ignoring the fact that alarms often propagate along specific routes, like lightpaths. In [81], a weighted alarm causal graph is built via the Hawkes process and conditional independence tests, followed by identifying the root alarms in each time window with the influence maximization analysis. Nevertheless, it could lack a principle of setting window size according to different network characters. Concurrently, the work in [8] aims at obtaining the alarm causal graph through the multivariate Hawkes process. It assumes that an alarm is triggered by other alarms from the alarm sequence and those alarms reported by its topological neighbors, but this method may suffer from high computation complexity with the expansion of network topology.

Pattern mining-based approaches offer the advantage of identifying hidden alarm patterns to generate alarm correlation rules. However, they require hyperparameter configuration based on prior knowledge, and rules established for one network state might not be applied to another. A pioneering knowledge acquisition system dubbed TASA [19][42] semi-automatically discovers alarm episode rules and association rules. Inspired by this seminal work, Wang et al. [69] derive parent-child rules, indicating the alarm causal relations from frequent itemsets using the PrefixSpan algorithm. However, identifying potential parent alarms is challenging as it leans heavily on the experience of domain experts, and these parent alarms can change based on network topology. Furthermore, [84] adds a preprocessing step to PrefixSpan, which involves calculating an association matrix to uncover more significant alarm patterns. This addition, though, might lead to a marked increase in complexity with longer alarm sequences.

Nowadays, machine learning-based approaches have become predominant in alarm correlation research [68]. However, these methods, tailored to a specific network state, may need to retrain their models when there are changes in the network topology or traffic distribution. Wang et al. [66][82] employ K-means and the backpropagation (BP) neural network to quantify the alarm importance in high-frequency chain alarm sets mined by the rule mining algorithm termed Apriori. However, they overlook the alarm location information. In [50], failure locations are inferred using the long short-term memory (LSTM) network and BP to find the mapping between the alarming devices and faulty ones. Still, their approach is best suited for small-scale static networks with a limited number of lightpaths. In [75], the researchers utilize the Hopfield neural network to address the multi-failure scenario. This becomes challenging when establishing relations between the suspected failures and alarms, since the alarms surge and spread throughout numerous boards in contemporary OTN backbones. Furthermore, Yu et al. [77] address the single-link failure localization problem by classifying the root alarms and non-root alarms. They tackle this using a deep belief network. However, their approach is specific to a certain

network state and might not be applicable to others.

Li et al. [46, 47, 45] utilize a constructed alarm knowledge graph (KG) to train a graph neural network (GNN) to determine the root alarm(s) and localize the single failure. However, the KG primarily incorporates static knowledge without dynamic factors such as the network topology and traffic distribution. Furthermore, this framework is extended in [22] which considers the network topology for addressing the multi-failure scenario, where the network status KG and alarm KG are integrated. However, this model requires retraining whenever the network topology changes. In addition, [35] attempts to use the pre-trained natural language processing model, specifically the Bidirectional Encoder Representations from Transformers (BERT), to identify the root alarm and its correlated alarms. However, BERT might struggle to distinguish alarms with similar semantics. In [85][76], a deep neural evolution network is employed to filter out the single-failure location from the set of suspected fault locations. However, in practical scenarios, preparing a possible fault location set for each alarm might be challenging. Babbar et al. [3] localize a single power attenuation failure using two classifiers based on the light gradient boosting machine. Nonetheless, the pre-defined sliding window size taken by the authors may not be suitable for other network environments. Lastly, [36] proposes the alarm propagation graph neural network to get the root alarms and eliminate the false repair orders via [84], Bayesian networks, and GNN, but their alarm correlation conditions are mostly summarized from expertise and not integrated as a universal metric.

Apart from the approaches using failure-alarm dependency relationships, failure localization can also be achieved via machine learning methods using other performance data. In [52], irregular wavelength selective switch (WSS) can be localized by inspecting power spectrum density (PSD) that presents different distortion degrees. In [55][56], the authors design a network-wide soft failure localization framework, where they lean on an artificial neural network-based approach using the data from streaming telemetry. In [5], the generalized signal-to-noise ratio (GSNR) is leveraged for failure localization by analyzing the evolution of GSNR over time, where the failure in WSSs could lead to the decrease of GSNR value.

Finally, we create a table in 1.1 that summarizes the applicable failure scenarios of our proposed approach and the counterparts. Obviously, most of the existing methods can simply be applied to single failure scenarios while some of them are feasible for multi-failure but non-region level failure. Note that the region-level failure represents that the number of failed boards is more than 1 and those failed boards correspond to a connected subgraph of the board-level network topology. Our proposed FACT-FL framework is the only one that works for region-level multi-failure cases.

Table 1.1: Summary of applicable failure scenarios of different methods.

Method name	Single failure			Multi-failure	
	Board level	Node level	Link level	Non-region-level	Region level
FACT-FL	✓	✓	✓	✓	✓
[50]	✓	✓	✓	✓	
[75]	✓	✓	✓	✓	
[77]			✓		
[46, 47, 45]	✓	✓	✓		
[22]	✓	✓	✓	✓	
[35]	✓	✓	✓	✓	
[85][76]	✓	✓	✓		
[3]	✓	✓	✓		
[36]	✓	✓	✓		

1.4 Research Objectives

Motivated by the exclusive demand for an effective failure localization approach, this thesis aims to propose a novel framework of board-level failure localization in the OTN optical layer, referred to as Failure-Alarm Correlation Tree based Failure Localization (FACT-FL). Its anticipated output is one or multiple FACT(s), where each FACT takes one failed board and its correlated alarms as the tree root and leaves, respectively. This research is designed to implement FACT-FL with different schemes summarized as follows:

- **FACT-FL-Heuristic**: an approach that trains a binary classifier for learning the correlation measure and heuristically creates the feasible FACT(s).
- **FACT-FL-Chain**: an approach that learns a set of binary classifiers for recognizing correlation chains with different order values and generates feasible FACT(s) by handling an integer linear programing (ILP) problem.
- **FACT-FL-GNN**: an approach that harnesses graph neural network (GNN) for assessing the edge weights of potential FACT(s) and obtains viable FACT(s) by solving an alternative simplified ILP.

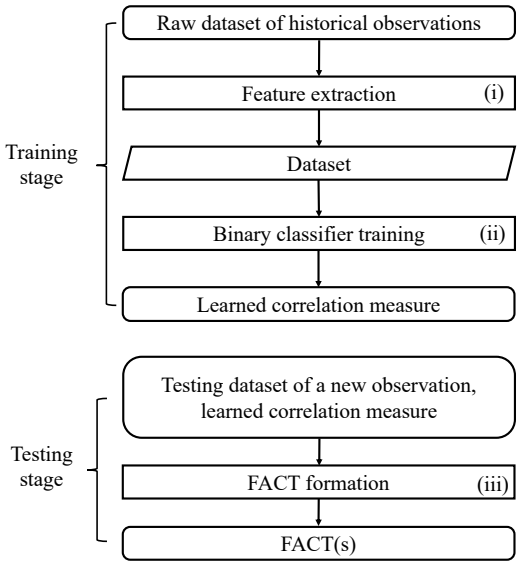


Figure 1.1: Flowchart of FACT-FL.

1.5 Thesis Structure

Fig. 1.1 demonstrates the flowchart of FACT-FL which consists of three functional blocks: feature extraction, binary classifier training, and FACT formation. We have realized FACT-FL through three approaches, named FACT-FL-Heuristic (Chapter 2), FACT-FL-Chain (Chapter 3), and FACT-FL-GNN (Chapter 4), respectively. Each approach adopts a different strategy to fulfill each functional block.

The rest of this thesis is organized as follows. Chapter 2 raises FACT-FL-Heuristic, a method that heuristically constructs FACT(s) grounded in the correlation measure learned by a binary classifier. Chapter 3 proposes FACT-FL-Chain, an approach that yields each FACT via selecting a suite of eligible correlation chains, which is achieved by addressing an ILP whose variables correspond to the chain candidates identified by the set of trained binary classifiers. Chapter 4 presents FACT-FL-GNN, which firstly utilizes GNN for evaluating the edge weights of latent FACTs induced by a collection of functional graphs (FGs), followed by framing a simplified ILP to construct the most likely FACT(s). Furthermore, extensive case studies are carried out in Chapter 2, 3, and 4 to compare the performance of proposed methodologies with that of the counterparts in terms of the metrics for evaluating the identified failed boards/root alarms. Finally, we summarize the thesis and discuss the future work in Chapter 5.

Chapter 2

FACT-FL-Heuristic

Failure localization serves as a key to an efficacious fault management plane in the Internet backbone. This chapter investigates a novel board-level failure localization framework, namely Failure-Alarm Correlation Tree based Failure Localization (FACT-FL), for achieving effective fault management in optical transport network (OTN). The FACT-FL is aimed at localizing failed device boards in the optical layer of OTN by correlating the failed boards and alarms. The outcome of FACT-FL is one or multiple FACT(s), where each FACT takes one failed board and its correlated alarms as the tree root and leaves, respectively. To implement FACT-FL, we present FACT-FL-Heuristic, an approach that heuristically creates the feasible FACT(s) using the correlation measure learned by a binary classifier. Notably, the proposed binary classifier is characterized by an intelligent feature extraction of historical correlations in dimensions of time, board/alarm attribute, network topology, and traffic distribution. Extensive case studies are conducted to demonstrate the advantages gained by FACT-FL-Heuristic in terms of its high precision, as well as the analysis of its performance due to various environmental turbulence such as network topology, traffic diversity, and noise alarms.

2.1 Introduction

Alarm correlation has been considered an effective approach to achieve the required precision in identifying dependency for each pair of collected alarms. With those dependencies, most dependent alarms can be removed such that the failure event(s) can be inferred/pinpointed with remarkably reduced complexity.

An example is given in Fig. 2.1 where five boards A, B, C, D, and E are connected by corresponding fiber pairs. As shown in Fig. 2.1(a), let the fiber cut event on the link from D to B be denoted as f_1, which is firstly detected by B, noted as an event f_2, and further triggers an alarm a_2 reported to the NMS. Upon f_2, B notifies its neighboring boards D and E as events f_3 and f_4, respectively, where the incurred alarms a_3 and a_4 are reported to the NMS accordingly. The entire failure-alarm propagation process is represented as a failure-alarm correlation tree (FACT) shown in Fig. 2.1(b). Note that although f_1 does not correspond to any alarm reported to the NMS, it is the root cause that requires a further inference to localize.

Another example is given in Fig. 2.1(c) where the failure on board D, noted as f_5, has even disabled its sensor and thus reported no alarm to the NMS. The failure event f_5 propagates to boards A, B, and C due to the commonly traversing traffic (i.e., an OTS, OMS, or OCH connection) that causes events f_6, f_7, and f_8, which further triggers alarms a_6, a_7, and a_8 reported to the NMS, respectively. At the same time, board E has event f_9 to occur due to the notification by B and reports an alarm a_9. The expected result of failure localization is by correlating the alarms a_6, a_7, a_8, and a_9, by which the NMS has to come up with the FACT as shown in Fig. 2.1(d) in spite of the fact that the root cause f_5 is completely "silent" throughout the whole alarm propagation and reporting process.

In addition to precision, the desired features of failure localization design include sufficient generality to various network environments and adaptability to the changing network status, including any possible variation in network topology and traffic distribution. Further, scalability should be pursued such that the NMS can swiftly identify the observed irregularity and launch the required reaction/restoration to the incurred damages.

Motivated by its importance and stringent requirements, this chapter introduces a novel failure localization framework in the optical layer of OTN, called Failure-Alarm Correlation Tree based Failure Localization (FACT-FL), aiming to explore various design dimensions for achieving all the desired features. FACT-FL is realized by the approach called FACT-FL-Heuristic, which heuristically creates the feasible FACT(s) using the correlation measure learned by a binary classifier. We firstly define *board instances* and *alarm instances*, which represent the status of each board and the collected alarms at the NMS during an observation window, respectively. By assuming that a functioning board can become faulty at most once at the beginning of an observation window, a board instance can be at most directly correlated with a set of alarm instances in time vicinity; while an alarm instance contains a number of features related to the reported alarm.

By taking each instance as a vertex, the correlation of an instance pair is nothing but the likelihood of the existence of an arc interconnecting the two vertices. As such

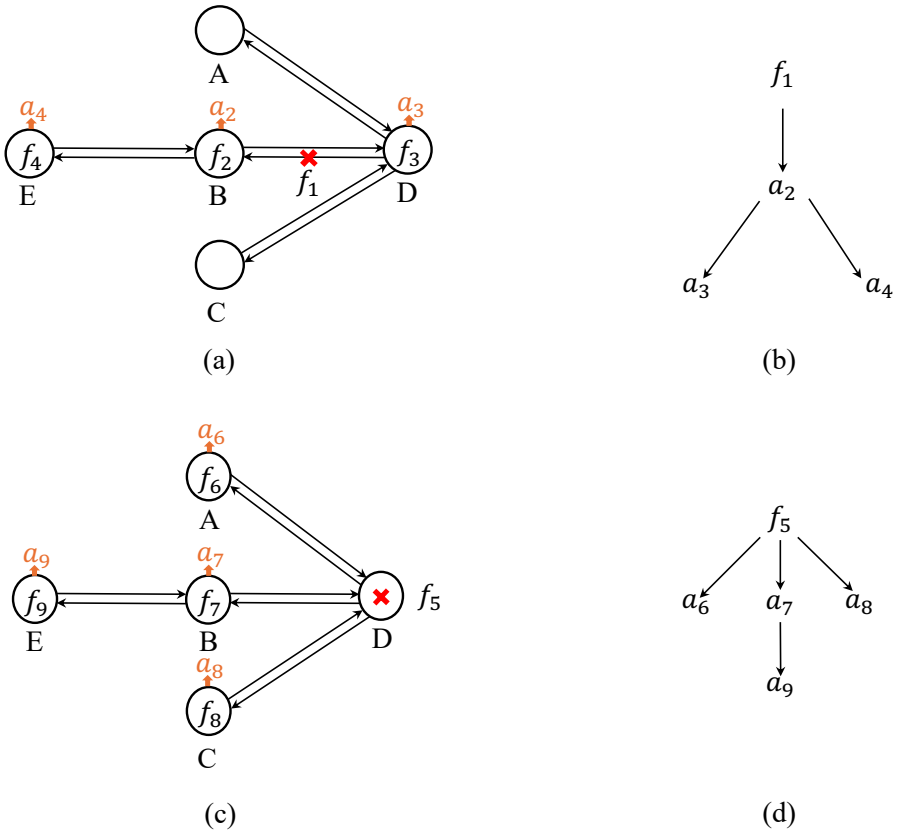


Figure 2.1: (a), (c) Examples of different failure events hitting the same network, where the red cross indicates the faulty fiber segment/board. (b), (d) Corresponding FACTs.

we investigate the instance correlation measurement using a machine learning approach by jointly considering the network topology and dynamic traffic distribution, where the trained binary classifier is migratable to any possible network environment with the same failure-alarm propagation rules. With all the labeled arcs, *FACT formation* can be exclusively completed by heuristically solving an integer linear programing (ILP) problem, where each FACT has a board instance as the tree root connecting to one or multiple alarm instances. The goal of the FACT formation is to cover all the alarms by the FACTs where each FACT demonstrates the complete alarm propagation process due to a faulty board.

The contributions of this chapter are summarized as follows.

- Investigate a novel failure localization approach, namely FACT-FL-Heuristic, which relies on a machine learning-based binary classifier and FACT formation modeling approach.
- Propose a novel binary classifier model, aiming to achieve the best generality and adaptation to the versatile network environment by taking both board instances and alarm instances as the input of the model.
- Introduce a novel FACT formation process for obtaining the most likely FACTs according to the given set of alarms.
- Conduct extensive case studies to verify the proposed FACT-FL-Heuristic approach and show that it can achieve effective and precise failure localization in the optical layer of OTN.

The rest of this chapter is organized as follows. Section 2.2 presents the system model, followed by the proposed FACT-FL-Heuristic approach in Section 2.3. Section 2.4 expounds on the case study setup and the results. Section 2.5 concludes this chapter.

2.2 OTN Failure-Alarm Propagation Model

In this study, two types of instances are defined. An *alarm instance* is denoted by a 5-tuple $a_i = \{t_i, h_i, b_i, m_i, r_i\}$, $\forall i \in \{1, \cdots, N\}$, where t_i is the occurrence time, h_i and b_i are the ID and type of the board that issues this instance, respectively. m_i is the alarm type and r_i is the layer information that can be either OTS, OMS, or OCH. A *board instance* is denoted by a 2-tuple $f_j = \{h_j, b_j\}$, $\forall j \in \{1, \cdots, M\}$, where h_j is the board ID and b_j is the board type. Note that the board instance can be any type of device board in the network

topology, including fiber segment, fiber interfact unit (FIU), optical amplifier (OA), optical transponder unit (OTU), optical multiplexer (OM), optical demultiplexer (OD), etc.

The correlation of two instances is evaluated between every pair of instances. An instance correlation between f_j and a_i could be due to the fact that the failure on board instance f_j triggers an alarm instance a_i, which is denoted as $f_j \rightarrow a_i$. Another possible scenario of instance correlation is between a_i and $a_{i'}$, where the alarm instance corresponding to a_i triggers another alarm instance $a_{i'}$, which is denoted as $a_i \rightarrow a_{i'}$.

In the setting of OTN, a board that has been hit by a failure/received a notification signal would change the status of board instance f_j/initiate alarm instance a_i, and this board could notify a remote board of reporting another alarm instance $a_{i'}$ to the NMS. The direction of alarm propagation can be divided into *forward propagation* (FP), *backward propagation* (BP), and *local notification* (LN). With FP (or BP), a board instance reports an alarm instance and sends a notification signal to downstream (or upstream) board instances that in turn emit corresponding alarm instances. LN, on the other hand, must be a single alarm instance reported by a board instance without any failure-alarm propagation process. Furthermore, we assume that the alarm propagation occurs along the shortest path between the source board and the destination board. Also note that the direction of alarm propagation can be used to determine the locations of the source board and destination board, which facilitates calculating the length of the shortest path between these two boards.

Accordingly, the failure-alarm propagation behavior in OTN is modeled into the following two generic types:

- **One2Many-Static-SameLayer/CrossLayer**: one failed board instance/alarm instance whose occurrence time is earlier triggers n alarm instances that happen later in the same/upper layer reported by no more than $n+1$ different boards in the way of FP, BP, or LN, respectively. The value of n is constant regardless of dynamic traffic distribution.

- **One2Many-Dynamic-SameLayer/CrossLayer**:
  one failed board instance/alarm instance that occurs earlier leads to n alarm instances that take place later in the same/upper layer emitted by at most $n+1$ various boards in the way of FP, BP, or LN, respectively. The value of n depends on the traffic distribution.

Due to the above characters, the OTN failure-alarm propagation process incurred by a failure event can be represented by one or multiple FACTs.

2.3 Proposed FACT-FL-Heuristic Approach

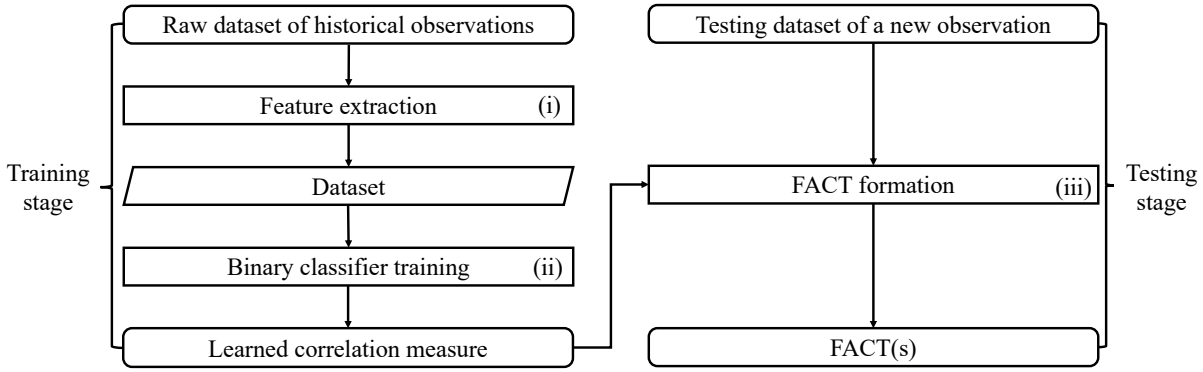


Figure 2.2: Flowchart of the proposed FACT-FL-Heuristic approach.

Fig. 2.2 shows the flowchart of the proposed FACT-FL-Heuristic approach that aims to construct the FACT(s) according to a set of alarms during an observation window. Given the raw dataset collected from the historical observations, our first step is to obtain attractive features from each instance pair as exhibited in (i) of Fig. 2.2. The details of feature extraction are given in 2.3.1. Then the obtained training dataset is used to train a binary classifier that learns the correlation measure of an instance pair, as demonstrated in (ii) of the flowchart whose details will be given in 2.3.2.

With the learned correlation measure, given a testing dataset collected by observing a network within an observation window P, all correlations among the instances can be explored through the FACT formation process as illustrated in (iii) of Fig. 2.2 that will be detailed in 2.3.3. Eventually, one or a set of FACTs corresponding to the given alarm set shall be obtained as the output of the proposed approach.

Additionally, Table 2.1 has summarized all notations used in this chapter for better readability.

2.3.1 Feature Extraction

The raw dataset is provided according to the historical data from carrier operators, which is collected within a set of observation windows denoted as $\mathcal{T} = \{T_1, \cdots, T_k, \cdots, T_K\}$, $\forall k \in \{1, \cdots, K\}$. During T_k we observed the network topology G_{T_k} and a set of lightpaths denoted as L_{T_k}. Here, G_{T_k} represents the interconnection of boards and each board could

Table 2.1: Summary of symbols and their definitions

Symbol	Definition
T_i	Observation window
G_{T_i}	Network topology in T_i
L_{T_i}	Traffic distribution in T_i
A_{T_i}	Alarm instance set in T_i
N_{T_i}	Number of alarm instances in F_{T_i}
F_{T_i}	Board instance set in T_i
M_{T_i}	Number of board instances in F_{T_i}
D_{T_i}	Set of all possible instance pairs in T_i
U_{T_i}	Set of ground-truth instance correlations in T_i
D	Training dataset

be a fiber segment or device whose failure would interrupt the traversing optical traffic flows. A total number of N_{T_k} alarm instances that have been sorted in ascending order of their occurrence time are denoted as $A_{T_k} = \{a_1, \cdots, a_{N_{T_k}}\}$. A total number of M_{T_k} board instances, denoted as $F_{T_k} = \{f_1, \cdots, f_{M_{T_k}}\}$, are obtained by considering all boards in G_{T_k}. Based on A_{T_k} and F_{T_k}, a set denoted as D_{T_k} with a size far smaller than $\frac{1}{2}[(N_{T_k} + M_{T_k})^2 - (N_{T_k} + M_{T_k})]$, contains all possible instance pairs. A set of ground-truth instance correlations denoted as $U_{T_k} = \{f_j \rightarrow a_i | i \in \{1, \cdots, N_{T_k}\}, j \in \{1, \cdots, M_{T_k}\}\} \bigcup \{a_i \rightarrow a_{i'} | i, i' \in \{1, \cdots, N_{T_k}\}\}$, is prepared. It is used for generating the binary label for each instance pair in D_{T_k} during the training process.

To characterize each instance pair $(v_i, v_j) \in D_{T_k}$, we create a feature space $\mathcal{H}$ in dimensions of time, board/alarm attribute, network topology, and traffic distribution. The feature vector of (v_i, v_j), denoted as $\mathcal{H}(v_i, v_j)$, is defined in (2.1):

$$\mathcal{H}(v_i, v_j) = [\Delta t_{i,j}, |l_{i,j}|, M(l_{i,j}), b_i, m_i, r_i, b_j, m_j, r_j, C(l_{i,j})], \tag{2.1}$$

where $\Delta t_{i,j}$ denotes the time gap of two instances, $|l_{i,j}|$ denotes the length of the shortest path from h_i to h_j in G_{T_k}, and $M(l_{i,j})$ counts the number of OMS connections traversed by the boards on the shortest path. b_i, m_i, r_i and b_j, m_j, r_j are the board type, instance type, and layer type of instance v_i and v_j, respectively. Finally, $C(l_{i,j})$ is a binary value that indicates whether h_i and h_j are traversed by a common OCH and can be obtained by checking L_{T_k}. Note that for the instance pair that contains a board instance, the missing features are complemented, where the occurrence time is set as the beginning moment of the observation window and an additional layer type is introduced.

By mapping each instance pair from D_{T_k} to $\mathcal{H}$, the transformed dataset $\mathcal{H}(D_{T_k})$ can be obtained. $\forall k \in \{1, \cdots, K\}$, the training dataset derived from all historical data, which is denoted as $D = \bigcup_{k=1}^{K} \mathcal{H}(D_{T_k})$, can be obtained for the subsequent binary classifier training.

2.3.2 DNN Architecture

The proposed binary classifier is used to evaluate the correlation of an instance pair in terms of the correlation measure. For this, a deep neural network (DNN) is employed whose architecture is illustrated in Fig. 2.3, where the dimension of each layer is given in parentheses. According to (2.1), $\mathcal{H}(v_i, v_j)$ consists of the numerical features including $\Delta t_{i,j}, |l_{i,j}|, M(l_{i,j})$ as well as the categorical features containing $b_i, m_i, r_i, b_j, m_j, r_j, C(l_{i,j})$. To combine these two types of input features, we map the categorical features of $\mathcal{H}(v_i, v_j)$ in continuous space by an embedding layer. Its output is concatenated with the numerical features of $\mathcal{H}(v_i, v_j)$ and fed into the subsequent fully-connected layers. Eventually, the model outputs the probability of instance correlation $v_i \to v_j$, denoted as $Pr\{v_i \to v_j\}$.

In addition, since most instance pairs are non-correlated, D is imbalanced. We adopt the resampling approach to build a balanced dataset through oversampling the minority class by random duplication [49][31].

2.3.3 FACT Formation

During an observation window P, the testing dataset of an arbitrary network state is provided, which incorporates the network topology G_P, a set of lightpaths L_P, and an alarm instance set A_P. The set of board instances is denoted as $F_P = \{f_1, \cdots, f_{M_P}\}$. As shown in Fig. 2.4, the goal of the proposed FACT formation is to construct one or multiple FACTs, each with a board instance as the tree root and some alarm instances as the leaf nodes, such that the alarm instances can be covered by the FACT(s) to the maximum extent. With the FACT(s), the required failure localization and alarm correlation can be achieved.

The following assumptions are held in the proposed FACT formation process. Firstly, the state of any board in F_P can only change at most once at the beginning of each observation window, i.e., either staying normal or switching from normal to failed. Once a board is failed, it stays in the failed state until the end of the observation window. This implies that a board instance may not correlate to any alarm instance and will not be taken into the FACT formation process. Secondly, all alarm instances in A_P are caused by one or multiple boards being failed at the starting moment of P. If there exists any

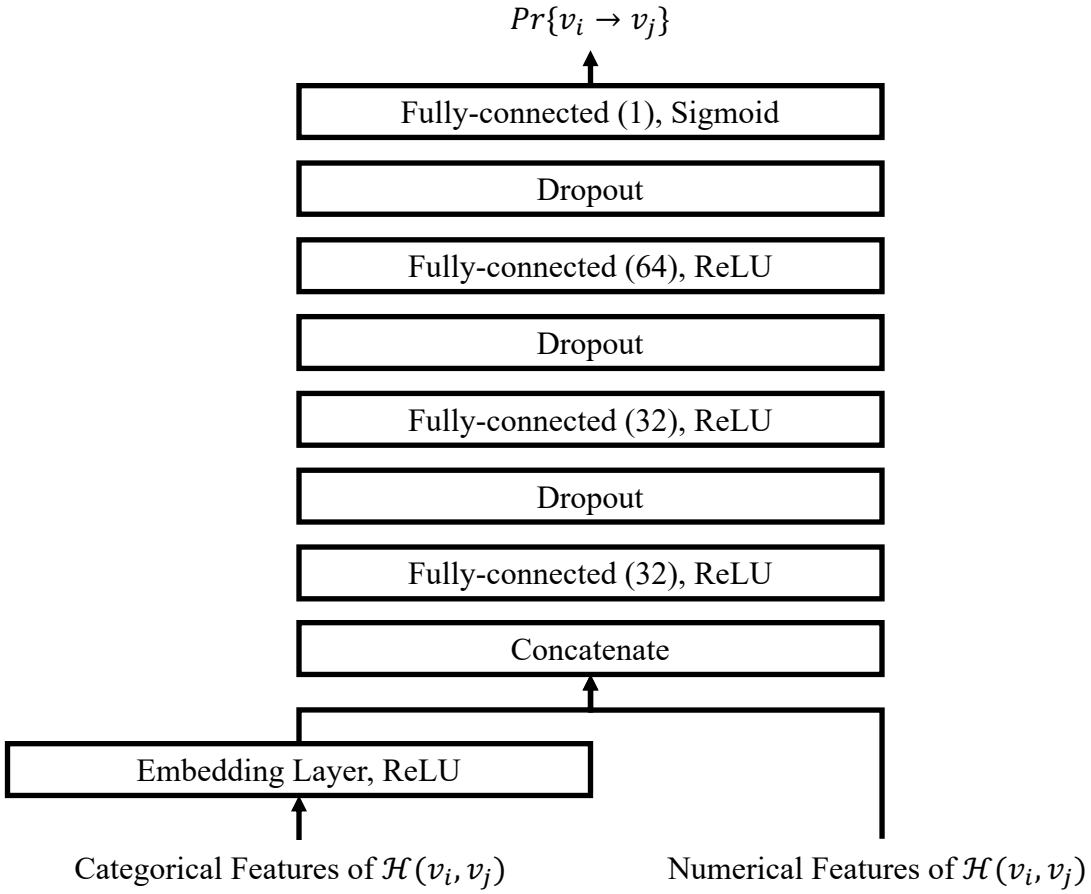


Figure 2.3: Architecture of DNN.

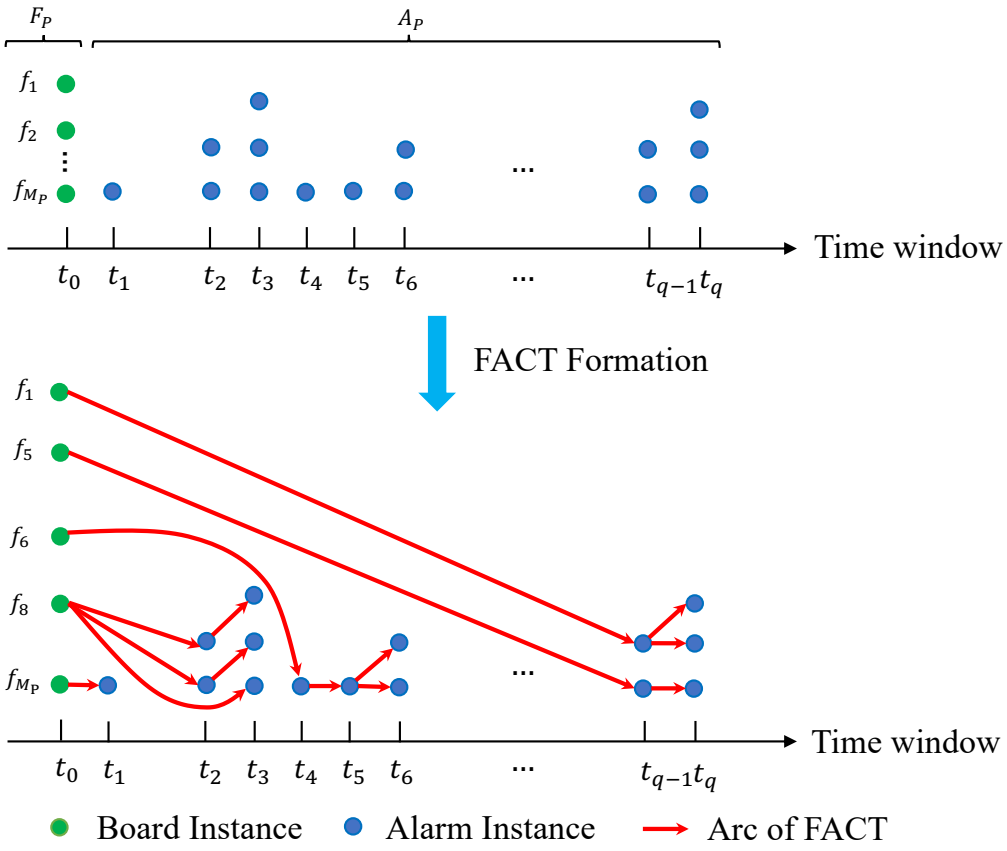


Figure 2.4: Illustration of the input and output of FACT formation.

alarm instance associated with a board that wasn't faulty at the beginning of P, it could be removed by examining the historical FACTs. Thirdly, due to the characters of failure-alarm propagation behavior in OTN, *One2One* and *One2Many* could happen whereas *Many2One* isn't allowed.

Based on the above assumptions, one or multiple FACTs constitute a directed forest denoted as $\mathcal{G}_P^I = (V_P^I, E_P^I)$, where V_P^I is the vertex set of instances containing the faulty board instances and all alarm instances, and E_P^I is the arc set covering all identified instance correlations.

To provision the search space for discovering $\mathcal{G}_P^I$, a weighted directed acyclic graph $\mathcal{G}_P$ is defined to incorporate all possible instance correlations, denoted as $\mathcal{G}_P = (V_P, E_P, W(E_P))$. $V_P = F_P \bigcup A_P$ is the vertex set of all board instances and alarm instances. $E_P = E_P^1 \bigcup E_P^2$ is the set of arcs, where $E_P^1 = F_P \times A_P$, $E_P^2 \subseteq A_P \times A_P$ are the sets of all possible instance correlations. The alarm instances in A_P have been sorted in ascending order of their occurrence time. E_P^2 can be obtained by considering all alarm instance pairs whose time gap is greater than 0. Whereas $W(E_P)$ is the set of non-negative weights for arcs in E_P. $\forall v_i, v_j \in V_P, (v_i, v_j) \in E_P, w_{ij} \in W(E_P)$ represents the cost of instance correlation $v_i \to v_j$ and it's denoted in (2.2):

$$w_{ij} = \begin{cases} 1 - Pr\{v_i \text{ is failed}\} \\ \quad \cdot Pr\{v_i \to v_j\}, & \text{if } (v_i, v_j) \in E_P^1, \\ 1 - Pr\{v_i \to v_j\}, & \text{if } (v_i, v_j) \in E_P^2, \end{cases} \tag{2.2}$$

where $Pr\{v_i \text{ is failed}\}$ is the probability that the board instance v_i becomes faulty. It can be determined by the probability density function of time-to-failure (TTF) of the corresponding board, which is estimated according to the historical failure events hitting this board. $Pr\{v_i \to v_j\}$ is the probability of instance correlation $v_i \to v_j$, which is calculated by the trained binary classifier.

2.3.4 Integer Linear Programing (ILP)

The problem of abstracting the best possible $\mathcal{G}_P^I$ from $\mathcal{G}_P$ can be formulated as an ILP given as follows:

$$\text{minimize} \quad \sum_{e \in E_P} w_e x_e + \sum_{u \in F_P} (1 - Pr\{u \text{ fails}\}) \cdot y_u \tag{2.3a}$$

$$\text{subject to} \quad \sum_{e \in \delta^-(u)} x_e = 1, \quad \forall u \in A_P, \tag{2.3b}$$

$$x_e \leq y_u, \quad \forall u \in F_P, \forall e \in \delta^+(u), \tag{2.3c}$$

$$x_e \in \{0, 1\}, \quad \forall e \in E_P, \tag{2.3d}$$

$$y_u \in \{0, 1\}, \quad \forall u \in F_P, \tag{2.3e}$$

where $\forall e \in E_P, w_e \in W(E_P), \forall u \in V_P, \delta^-(u), \delta^+(u)$ are the sets of all incoming arcs and outgoing arcs of vertex u, respectively. Two binary variables x_e, y_u are defined, where x_e takes 1 if the instance correlation arc e is chosen by $\mathcal{G}_P^I$ and 0 otherwise; while y_u takes 1 if the board instance vertex u is selected as a tree root in $\mathcal{G}_P^I$ and 0 otherwise. The objective function (2.3a) aims to find the $\mathcal{G}_P^I$ that minimizes the total cost of selected instance correlations and board instances. Constraint (2.3b) indicates that for each alarm instance vertex, only one incoming arc is selected by $\mathcal{G}_P^I$. This guarantees that all alarm instances are traversed by $\mathcal{G}_P^I$ and the in-degree of each alarm instance vertex must be one, which satisfies $\mathcal{G}_P^I$'s property of being a directed forest. Constraint (2.3c) implies that for each board instance, if any one of its outgoing arcs is selected then this board instance vertex must be chosen as a tree root.

By solving the above ILP, the anticipated FACT(s) $\mathcal{G}_P^I$ can be obtained. The root nodes in $\mathcal{G}_P^I$ are faulty boards in the observed network state, where the failure localization is accomplished by checking their location information. Also, $\mathcal{G}_P^I$ elaborates all correlations among the faulty boards and alarms.

Obviously, solving the above ILP model could be subject to intolerably long computation time. Thus, a heuristic scheme is developed to come up with feasible solutions.

2.3.5 Heuristic Algorithm

Fig. 2.5 demonstrates the flowchart of the proposed heuristic algorithm that aims to construct the feasible FACT(s) $\mathcal{G}_P^I$. Given the set of instances V_P and its corresponding instance pair set E_P, we can obtain the set of arc weights $W(E_P)$ by passing through each

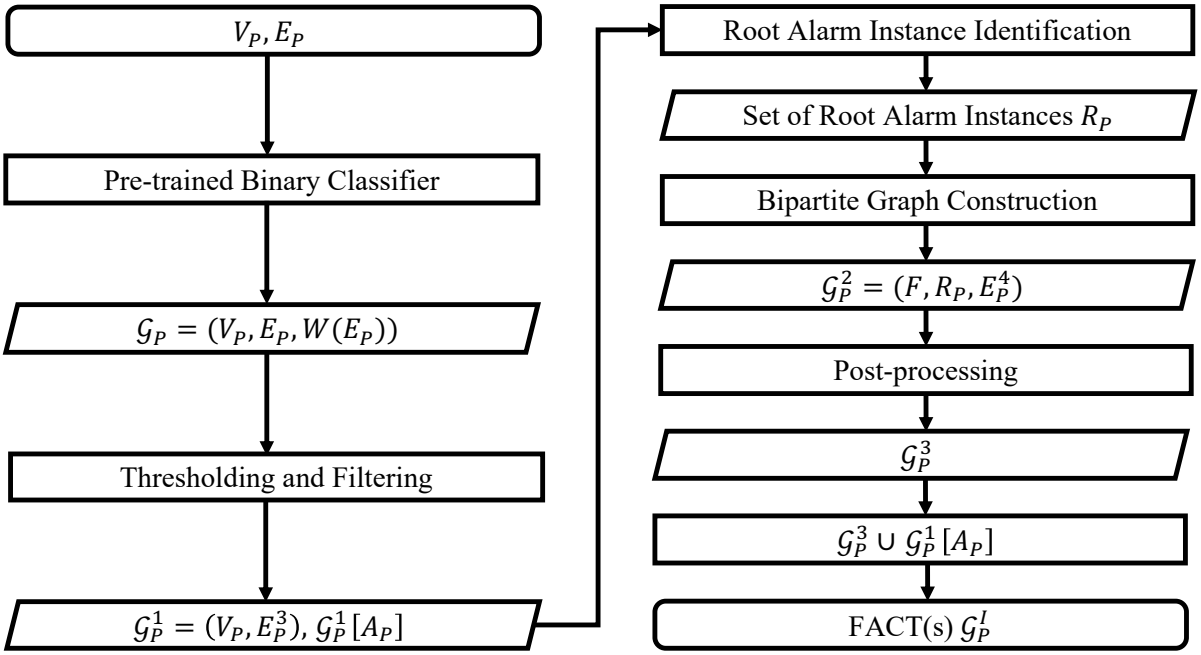


Figure 2.5: Flowchart of the proposed heuristic algorithm for FACT formation.

instance pair from E_P into the pre-trained binary classifier. For simplicity, we assume that each board instance is equally likely to be faulty, which implies that the cost of choosing a board instance can be ignored by setting $Pr\{u \text{ fails}\}$ as 1 in (2.3a).

To remove unreliable instance correlations in $\mathcal{G}_P$, the arcs whose weight in $W(E_P)$ is greater than 0.5 are discarded. Also, due to the characters of failure-alarm propagation behavior, for each alarm instance, there is at most one incoming arc whose tail vertex belongs to the alarm instance, where we only reserve one arc with the minimum weight if there are multiple qualified incoming arcs. Hence, $\mathcal{G}_P$ is reduced to be $\mathcal{G}_P^1 = (V_P, E_P^3)$. Meanwhile, the subgraph of $\mathcal{G}_P^1$ induced by A_P has become a directed forest, which represents all instance correlations formed by alarm instances and it's denoted as $\mathcal{G}_P^1[A_P]$.

Furthermore, the set of root alarm instances, denoted as $R_P \subseteq A_P$, can be identified as the alarm instance whose in-degree in $\mathcal{G}_P^1$ is non-zero and all its incoming arcs are initiated by board instances. For each root alarm instance $r_i \in R_P$, the corresponding set of board instances F_i and the set of associated alarm instances A_i can be determined by $\mathcal{G}_P^1$, where F_i contains all board instances that connect to r_i and A_i aggregates all alarm instances that are reachable from r_i. Based on the attributes of r_i and all alarm instances in A_i, the board instance whose confidence not only surpasses 0.5 but also is the highest one

could be chosen according to the mined association rules between the faulty board and its corresponding alarm types in the raw dataset and all other board instances are eliminated from F_i. In addition, the set of board instances F is acquired by taking the union of all F_i's.

To reflect the relationship between the set of board instances and root alarm instances, we define a directed bipartite graph $\mathcal{G}_P^2 = (F, R_P, E_P^4)$, where $E_P^4 \subseteq E_P^3$ is the arc set that represents all instance correlations between the board instances in F and root alarm instances in R_P. We will post-process it and denote the output as $\mathcal{G}_P^3$, which aims to cover the root alarm instances to the maximum extent by choosing the least number of board instances as the faulty boards. For each component in $\mathcal{G}_P^2$, we iteratively select one board instance with the maximum out-degree and all of its outgoing arcs until all root alarm instances have been explained by the corresponding board instances.

Eventually, the feasible FACT(s) $\mathcal{G}_P^I$ shall be obtained by taking the union of $\mathcal{G}_P^3$ and $\mathcal{G}_P^1[A_P]$. Note that there could exist more than one $\mathcal{G}_P^3$ for the same $\mathcal{G}_P^2$, leading to multiple $\mathcal{G}_P^I$'s, where we will take the union of all those solutions in case missing any possible faulty board.

2.4 Case Studies

Extensive case studies are conducted to verify the proposed FACT-FL-Heuristic method in OTN and compare it with a number of counterparts. An OTN simulator [44] is firstly developed to generate ground-truth alarms and the resultant FACTs $\mathcal{G}_P^T$ according to the given failure event, failure-alarm propagation rule database, as well as G_P and L_P during the observation window P. The FACTs produced by the proposed FACT-FL-Heuristic approach are denoted as $\mathcal{G}_P^E$. Currently, the relational rule database contains 39 rules which incorporate 65 instance correlation types formed by 20 failure types, 26 alarm types, and 16 board types. Without loss of generality, each failure event will independently hit a board, thereby affecting the traversing optical flows.

The goal of this case study includes the following two aspects:

- Evaluate the performance of the trained binary classifier on the training/validation set.
- Verify the generality and migratability of the proposed FACT-FL-Heuristic by comparing its performance with that of the counterparts on the testing datasets of single-board failure in the following network environments:

- use the same network topology and lightpath setting as the raw dataset;
- change the number of lightpaths for the given network topology;
- change the size of network topology for the given number of lightpaths;
- change the ratio of the number of noise alarms to that of true alarms.

The state-of-the-art counterparts considered in this case study include BP[50], LSTM[50], GNN[46, 47, 45], Transformer[35], and convolutional neural network (CNN)[77].

2.4.1 Setup

Raw Dataset

We generate the raw dataset by initiating a set of independent single-board failure events, where each of them in turn hits one board in the given network topology. The length of each observation window T_i is 1 min. The network topology G_{T_i} is characterized by S_i, $|F_{T_i}|$, deg_i, which are the number of nodes, the number of board instances, and the board-level average degree, respectively. Whereas the set of lightpaths L_{T_i} is described by $|L_{T_i}|$ and $|\overline{l_i}|$ that indicate the number of lightpaths and the average number of boards traversed by each lightpath. The setting of network topology and lightpath is consistent in all observation windows, where $\forall i \in [1, 561], S_i = 15, |F_{T_i}| = 561, deg_i = 2.48, |L_{T_i}| = 40$, and $|\overline{l_i}| = 14$.

Training Dataset

The raw dataset contains 7365 alarms, leading to a training dataset of size 2726247, where the ratio of positive samples to negative samples is 4970:2721277. We set a 64%, 16%, and 20% split for training, validation and test sets. For the numerical features, min-max scaling is applied for normalization. We adopt the binary cross-entropy loss function, which is optimized with Adam at a learning rate of 0.001. The batch size and the number of epochs are set to 450 and 100. Also, the technique of early stopping [7] is applied to reduce overfitting, which monitors the value of the area under the curve (AUC) on the validation set in each epoch.

AI Architectures of the Counterparts

The network architectures of the counterparts are briefly described as follows. For BP and LSTM, two networks of $233 \times 64 \times 32 \times 561$ and $233 \times 64 \times 561$ are constructed. For training the GNN, an alarm knowledge graph with 43 entity nodes is built based on the failure-alarm propagation rule database. The CNN model consists of 24 input layer units as well as 3 hidden layers whose number of neurons are 256, 128, and 32, where the kernel size in each hidden layer is 2×2. The Transformer encodes the alarm context with a 768-dimension vector and sets the length of each alarm transaction to 5.

2.4.2 Performance Metrics

The comparison between $\mathcal{G}_P^E$ and $\mathcal{G}_P^T$ is accomplished via three parts.

Metrics for Root Alarm Identification

Firstly, we evaluate the results of identified root alarms in terms of *precision(R)*, *recall(R)*, and *accuracy(R)*, which are defined as follows:

$$\begin{aligned} precision(R) &= \frac{NC_{ra,E}}{NT_{ra,E}}, \\ recall(R) &= \frac{NC_{ra,E}}{NT_{ra,T}}, \\ accuracy(R) &= \frac{NC_{a,E}}{|A_P|}, \end{aligned} \tag{2.4}$$

where $NC_{a,E}$ is the number of correctly inferred root and non-root alarm instances according to $\mathcal{G}_P^E$; $|A_P|$ is the size of alarm instance set A_P; $NC_{ra,E}$ is the number of correctly inferred root alarm instances in $\mathcal{G}_P^E$; $NT_{ra,E}$, $NT_{ra,T}$ are the number of root alarm instances in $\mathcal{G}_P^E$, $\mathcal{G}_P^T$, respectively.

Metrics for Failed Board Identification

Secondly, the performance of failure localization in terms of how precisely/accurately the failed boards can be identified is assessed by comparing the faulty boards in $\mathcal{G}_P^E$ with that

in $\mathcal{G}_P^T$, which is evaluated via *precision(F)*, *recall(F)*, and *accuracy(F)*, as given by:

$$\begin{aligned} precision(F) &= \frac{NC_{root,E}}{NT_{root,E}}, \\ recall(F) &= \frac{NC_{root,E}}{NT_{root,T}}, \\ accuracy(F) &= \frac{NC_F}{|F_P|}, \end{aligned} \tag{2.5}$$

where NC_F is the number of correctly inferred functioning and faulty boards according to $\mathcal{G}_P^E$; $|F_P|$ is the total number of boards in G_P; $NC_{root,E}$ is the number of correctly inferred faulty boards in $\mathcal{G}_P^E$; $NT_{root,E}, NT_{root,T}$ are the number of faulty boards in $\mathcal{G}_P^E$, $\mathcal{G}_P^T$, respectively.

Metrics for Alarm Instance Correlation

Thirdly, we evaluate the quality of alarm instance correlations in FACTs in terms of *recall(A)* and *accuracy(A)*, which are defined by:

$$\begin{aligned} recall(A) &= \frac{NC_{arc,E}}{NT_{arc,T}}, \\ accuracy(A) &= \frac{NC_A}{|D_P^A|}, \end{aligned} \tag{2.6}$$

where NC_A is the number of correctly labeled alarm instance pairs according to $\mathcal{G}_P^E$; $|D_P^A|$ is the size of alarm instance pair set D_P^A; $NC_{arc,E}$ is the number of correctly inferred alarm instance correlations in $\mathcal{G}_P^E$ and $NT_{arc,T}$ is the number of alarm instance correlations in $\mathcal{G}_P^T$.

2.4.3 Results

Binary Classifier

The training procedure of DNN is demonstrated in Fig. 2.6. The loss converges to 0.0087/0.014 and accuracy reaches 99.85%/99.72% after 15 epochs on the training/validation set.

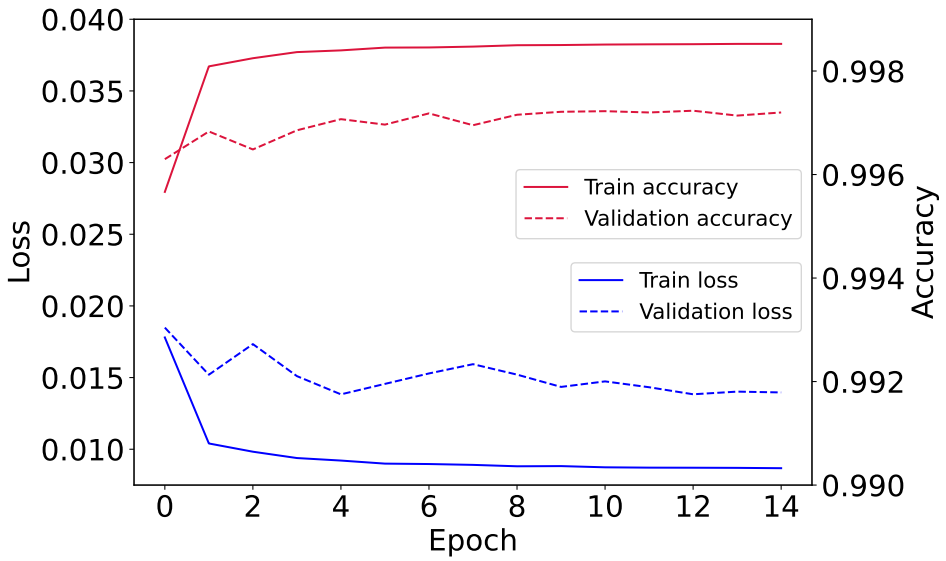


Figure 2.6: DNN performance on the training and validation set.

Table 2.2: Performance comparison among different schemes under the network environment of the raw dataset

Metrics / Schemes	# of trainable parameters	$precision(R)$	$recall(R)$	$accuracy(R)$
FACT-FL-Heuristic (IC-FD)	5585	1	0.9666	0.9975
CNN	150531	0.895	0.9666	0.9627
Transformer	2521716	0.2153	0.2666	0.7156
Metrics / Schemes	# of trainable parameters	$precision(F)$	$recall(F)$	$accuracy(F)$
FACT-FL-Heuristic (IC-FD)	5585	0.95	1	0.9998
BP	35569	0.85	0.9	0.9994
LSTM	112753	0.85	1	0.9994
GNN	51999	0.5833	0.7	0.7

FACT Formation

Firstly, we adopt the network setting of the raw dataset and conduct 10 independent single-board failure experiments. The average performance results of various schemes are summarized in Table 2.2. Apparently, the FACT-FL-Heuristic achieves a significant advantage in terms of all metrics regarding root alarm instance/failed board identification against the counterparts at the cost of the least number of parameters.

Furthermore, we verify the migratability of FACT-FL-Heuristic where the model is trained by using the raw data from an initial setting while being applied to some other network scenarios with different topologies and traffic distributions. On one hand, given a network topology whose $S_i = 14, |F_{P_i}| = 3470, deg_i = 2.08$, we vary the number of lightpaths $|L_{P_i}|$ from 50 to 500. On the other hand, we fix $|L_{P_i}| = 200, |\overline{l_i}| = 16$ and change the size of network topology, where $S_i \in [10, 37], |F_{P_i}| \in [1074, 1839], deg_i \in [2.1, 2.32]$. As shown in Fig. 2.7 and Fig. 2.8, the FACT-FL-Heuristic performs the best in root alarm identification among those three methods, where its $accuracy(R)$, $recall(R)$ stabilize above 97%, 90% and its $precision(R)$ remains above 93%. However, the performance by the CNN model is significantly degraded under certain values of $|L_{P_i}|/|F_{P_i}|$ and that by Transformer behaves even worse, which shows that they can't stably capture root alarms when the spatial relation and traffic distribution are completely ignored. Further as depicted in Fig. 2.10 and Fig. 2.11, $recall(F)$ of FACT-FL-Heuristic maintains 1 that implies no true faulty boards are missed, which isn't accomplished by LSTM and BP even if their $accuracy(F)$ and $precision(F)$ are similar to that of FACT-FL-Heuristic. Whereas the performance of GNN exhibits high fluctuation due to merely learning the mapping between the failure type and alarm type. Also, as displayed in Fig. 2.13 and Fig. 2.14, most alarm instance correlations are successfully identified by FACT-FL-Heuristic. Note that since the location information of faulty board/alarm varies with different network environments, all AI models taken by the counterparts need to be retrained as long as there is any change with the network topology/traffic distribution, whereas the FACT-FL-Heuristic only needs to be trained once with the alarm data collected from any given network state(s) but it shows the best migratability among all counterparts.

Finally, we test the anti-noise capability of the FACT-FL-Heuristic by introducing some noise alarms on top of the true alarms due to a single-board failure event. As shown in Fig. 2.9, CNN and Transformer suffer serious performance degradation in detecting root alarms as the ratio of noise alarms continues to increase, whereas FACT-FL-Heuristic demonstrates a good capability in overcoming the noises thanks to its additional consideration of the spatial relations among the received alarms. Similarly as illustrated in Fig. 2.12, FACT-FL-Heuristic can keep steady performance in faulty board identification

when encountering the noises while all the counterparts are subject to significant degradation.

Note that the total processing time of FACT-FL-Heuristic is proportional to the number of examined instance pairs each taking about 4 ms to handle.

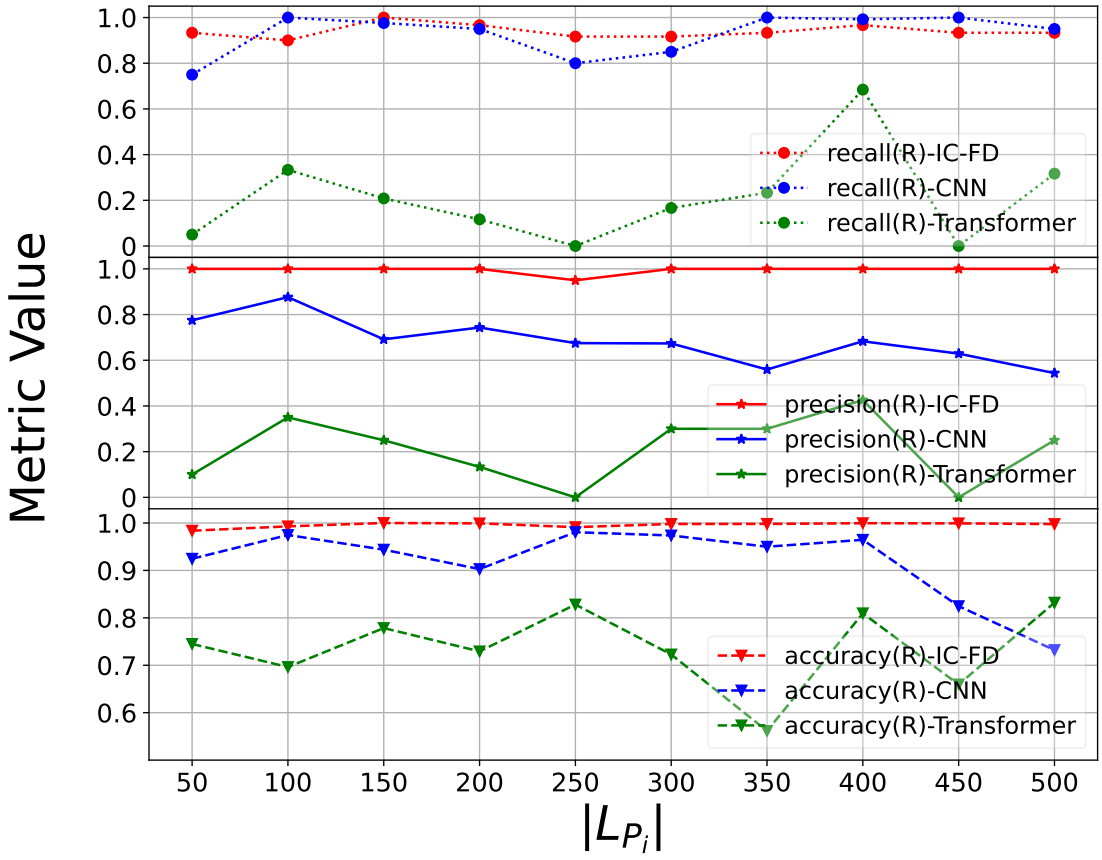


Figure 2.7: Performance result of root alarm metrics when changing $|L_{P_i}|$.

Complexity Analysis

We assume that during an OW P, the board set $F_P = \{b_1, \cdots, b_{M_P}\}$ and alarm set $A_P = \{a_1, \cdots, a_{N_P}\}$ are provided. The complexity of FACT-FL-Heuristic mainly derives

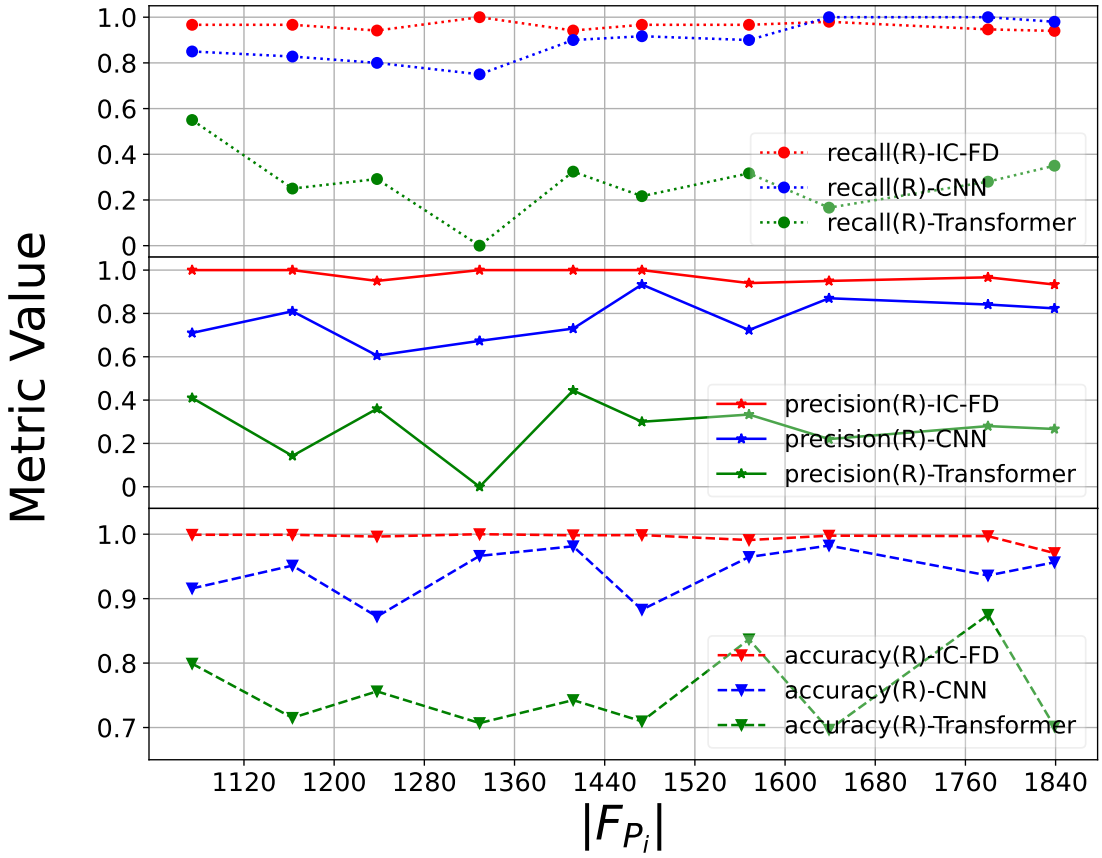


Figure 2.8: Performance result of root alarm metrics when changing $|F_{P_i}|$.

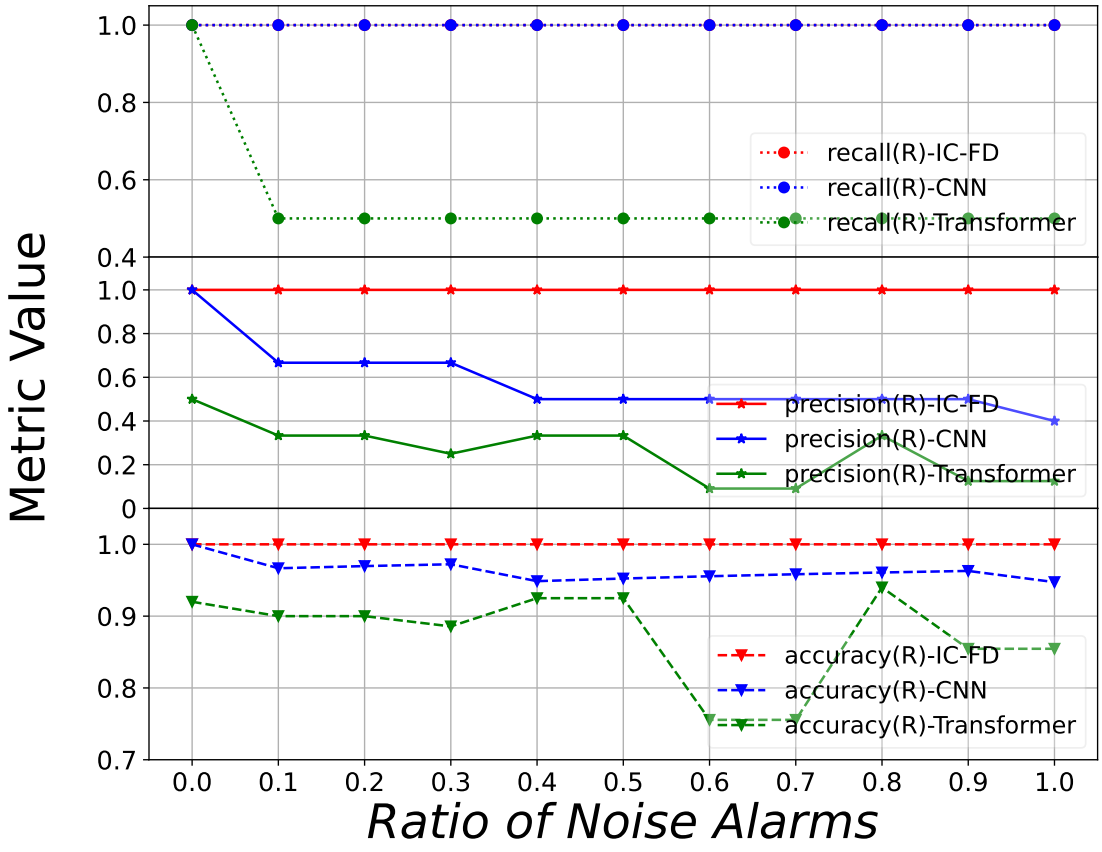


Figure 2.9: Performance result of root alarm metrics when adding noise alarms.

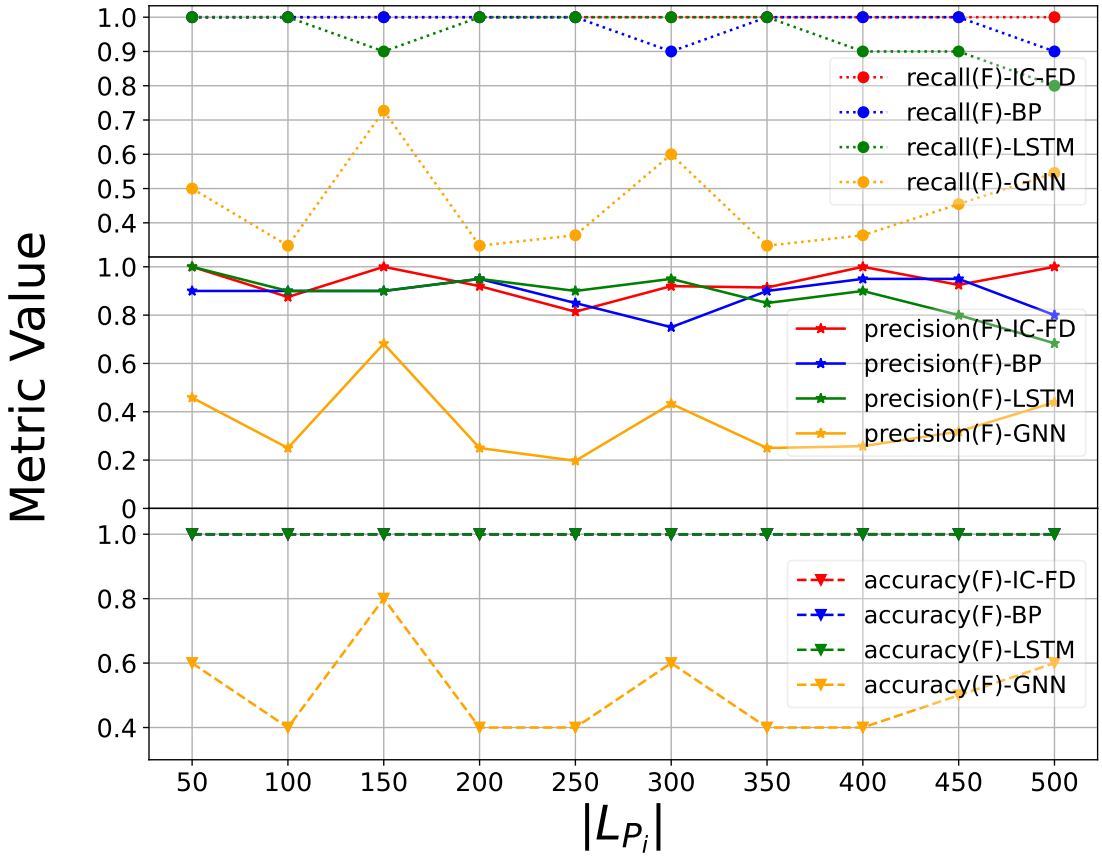


Figure 2.10: Performance result of failed board metrics when changing $|L_{P_i}|$.

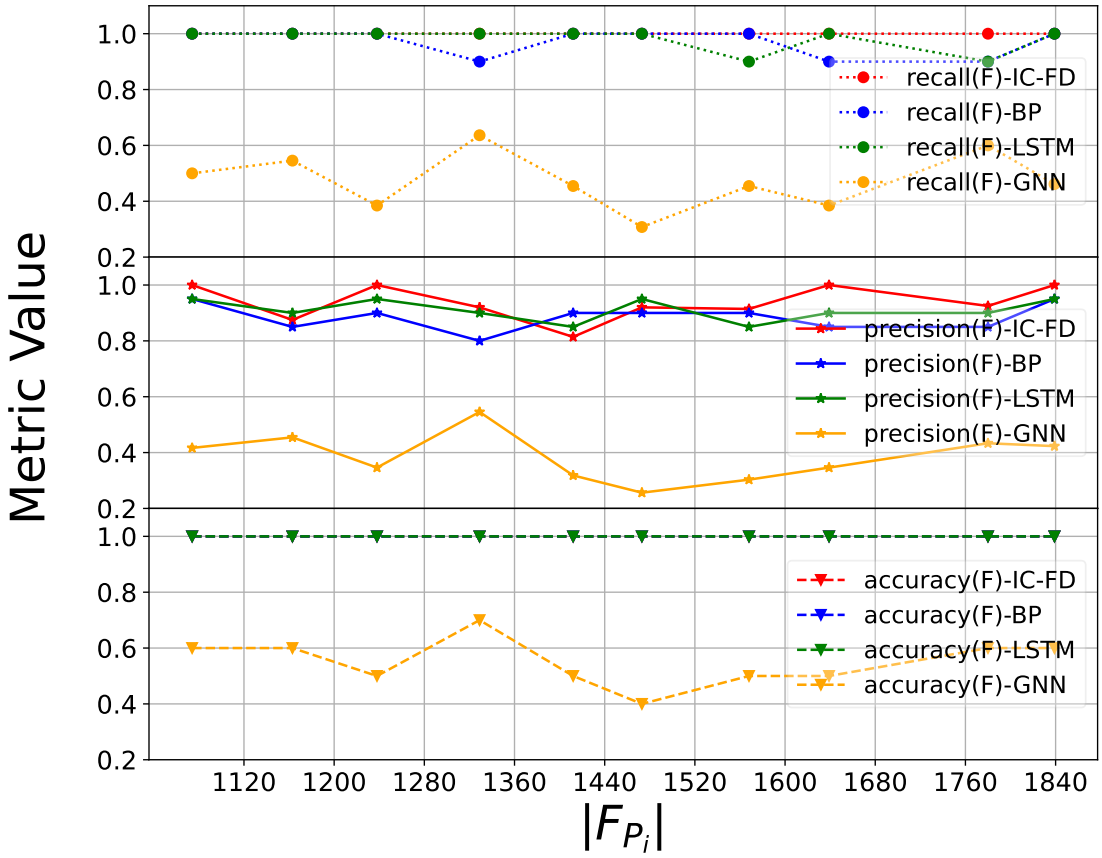


Figure 2.11: Performance result of failed board metrics when changing $|F_{P_i}|$.

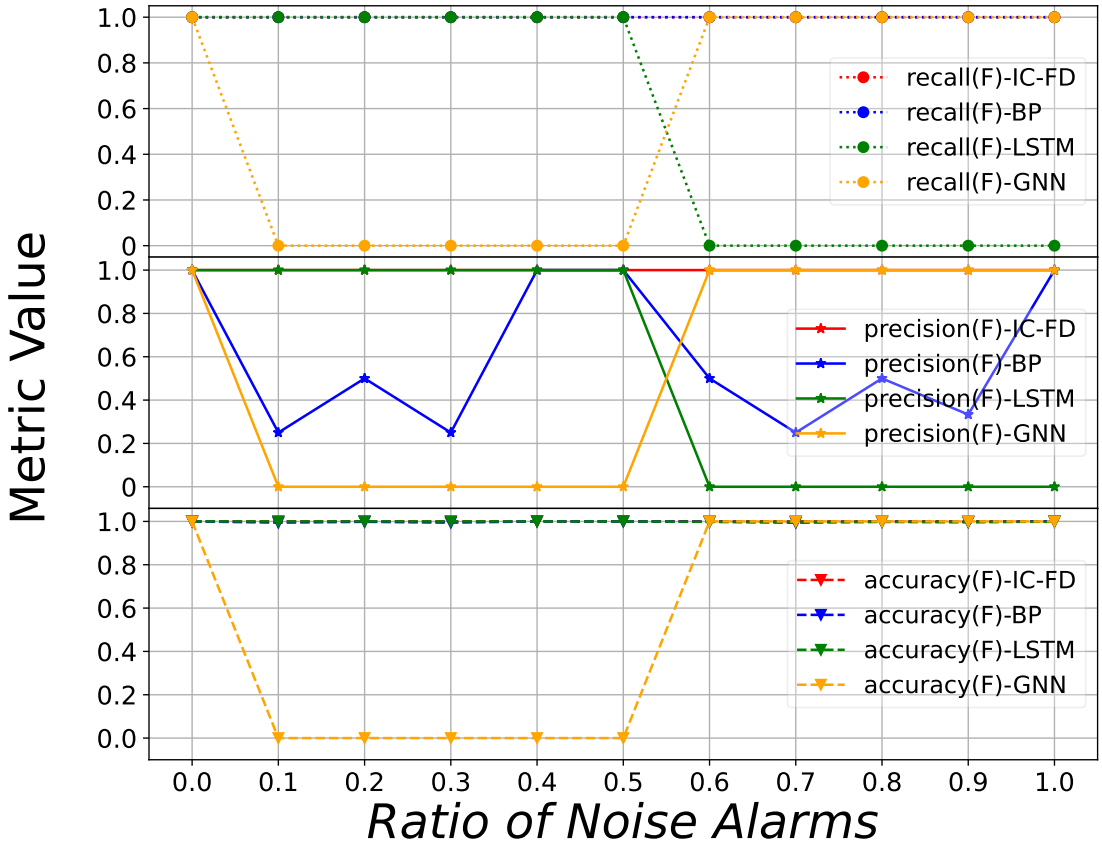


Figure 2.12: Performance result of failed board metrics when adding noise alarms.

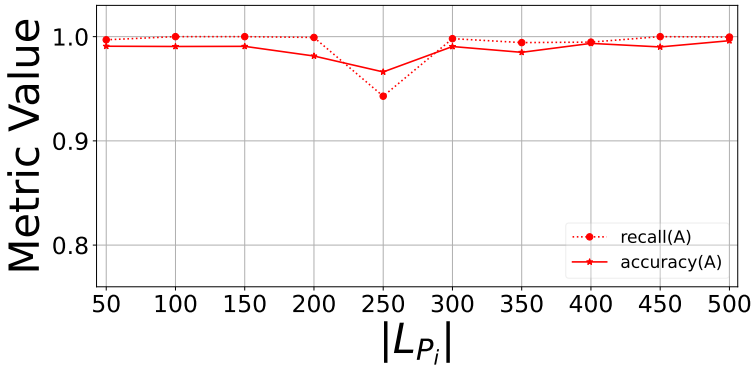


Figure 2.13: Performance result of alarm instance correlation metrics when changing $|L_{P_i}|$.

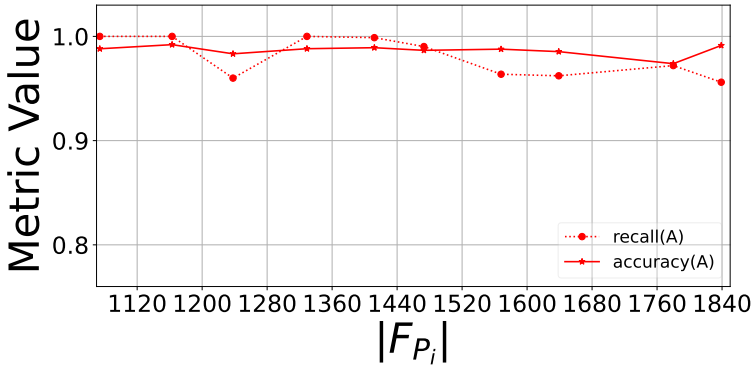


Figure 2.14: Performance result of alarm instance correlation metrics when changing $|F_{P_i}|$.

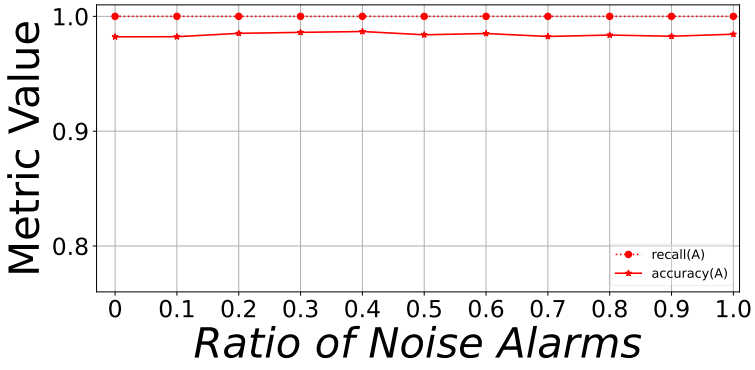


Figure 2.15: Performance result of alarm instance correlation metrics when adding noise alarms.

from the FACT formation procedure that uses the pre-trained binary classifier to examine all possible instance pairs, resulting in the complexity of $\mathcal{O}(M_P \times N_P + {N_P}^2)$.

2.5 Conclusion

This chapter introduced a novel failure localization and alarm analysis scheme in the optical layer of OTN, called FACT-FL-Heuristic, aiming to identify the affected boards due to a failure event with high precision. The proposed FACT-FL-Heuristic is characterized by a suite of novel modeling approaches. Firstly, we have included both the alarm instances and board instances in the correlation model to perform localization of the failed boards directly instead of merely obtaining the root alarms. Secondly, we developed a novel DNN-based binary classifier along with various features that consider all those static and dynamic network parameters, aiming to achieve sufficient generality and migratability for various network environments. Thirdly, the FACT can effectively describe the correlation between the alarms and the faulty boards, and the FACT formation process serves as a graceful solution that can swiftly come up with high-quality results.

Extensive case studies were conducted to verify the feasibility and performance of the proposed method and modeling approaches. Compared with its counterparts, the proposed FACT-FL-Heuristic scheme can adapt to versatile network environments (i.e., change of network topology, traffic distribution, or adding noise alarms) and achieve superb and stable performance in root alarm and failed board identification in terms of precision, recall, and accuracy.

Chapter 3

FACT-FL-Chain

This chapter presents a novel approach to achieving the FACT-FL framework, called FACT-FL-Chain. Specifically, an FACT consists of a suite of k^{th} order Failure-Alarm Correlation Chains (k-FACCs) with different order values of k. Each k-FACC indicates the chain-like correlation established by k alarms due to one common failed board. To identify all previously undetected k-FACCs, a set of binary classifiers is trained that characterizes each k-FACC from various dimensions, including the time, network topology, traffic distribution, and board/alarm attributes. Eventually, an integer linear programing (ILP) problem is formulated to extract the most likely FACT(s) from those k-FACCs. Extensive case studies demonstrate the superior results of FACT-FL-Chain in terms of metrics evaluating the identified failed boards and root alarms. We also analyze its performance under different maximum order values of k and environmental changes, including diverse failure scenarios, network topologies, traffic distributions, and noise alarms.

3.1 Introduction

Internet optical backbones have evolved remarkably over the past few decades to support higher bandwidth, expand geographical coverage, and increase heterogeneity. This adaptability accommodates the multi-service and multi-tenant environment. Optical transport network (OTN) has been standardized by ITU-T as a digital wrapper. This wrapper multiplexes services over a single lightpath for transparent transport [29], serving as the foundation for the rapid advancement of modern Internet backbones. The OTN optical domain adopts a 3-layer structure comprising the optical channel (OCH), optical multiplex section (OMS), and optical transport section (OTS) [9]. As illustrated in Fig. 3.1, each

OCH (lightpath) goes through a set of device boards that traverse one or multiple OMSs. Concretely, an OMS spans an extensive number of cascaded OTSs, which incorporates a variety of board types, including the optical transponder unit (OTU), optical multiplexer (OM), optical demultiplexer (OD), optical amplifier (OA), fiber interface unit (FIU), fiber segment, etc [27]. To facilitate monitoring performance, the OTN control plane exploits electrical and optical layer overheads for exchanging all types of maintenance signals between numerous boards. This enables each board to generate corresponding alarms in response to any failure event detected by its built-in sensor. For instance, if an FIU board identifies a failure in its upstream fiber segment, it will report an alarm to the network management system (NMS). Similarly, an OTU board will report an alarm to the NMS if it detects any anomaly that impacts the lightpath quality.

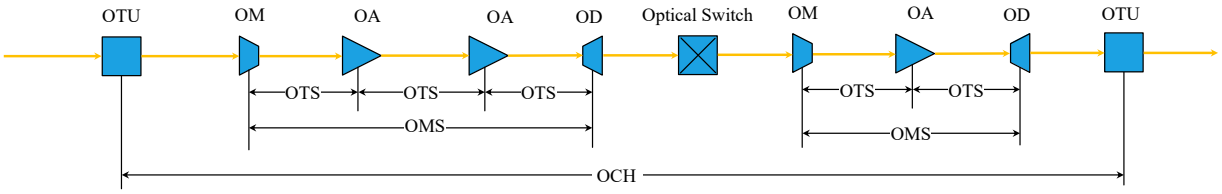


Figure 3.1: Relationships between OTN layers in a lightpath.

In most cases, a failure event may impact one or multiple boards. These boards could be scattered throughout the network or connected within a specific region, and this can unexpectedly disrupt countless lightpaths. Other boards along those interrupted lightpaths can also detect these disruptions. Moreover, a board might raise an alarm and report it to the NMS not just because of a detected failure event but also in response to a notification alarm from another remote board. Consequently, a failure event can lead to an alarm flood, significantly increasing the complexity of alarm analysis and failure localization—especially in network domains with a vast number of boards, such as the OTN backbones.

Alarm correlation is widely recognized as an effective method to identify dependencies among alarms. The goal is to discard as many dependent alarms as possible, thereby simplifying the complexity of failure localization.

An illustrative example is shown in Fig. 3.2a. The fiber segment cut, represented by f_1, is firstly detected by its downstream board B that issues alarm type a_1. On one hand, B further notifies its upstream board D via alarm type a_2. On the other hand, the effects of f_1 are detected along the lightpath by all other boards traversed by this path, successively triggering alarm types a_3, a_4, and a_5. Note that all alarms are subsequently reported to the NMS.

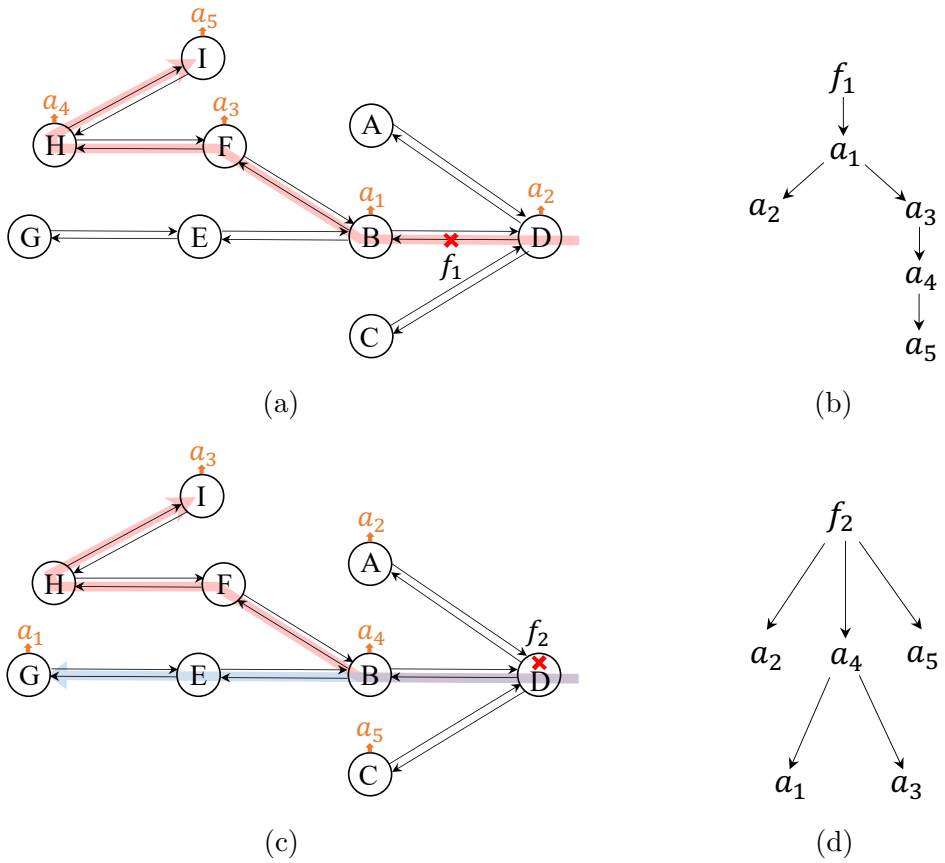


Figure 3.2: (a), (c) Illustration of two single-board failure events, where $A, B, \ldots, I$ are boards, red and blue arrows are lightpaths, $a_1, \ldots, a_5$ are alarm types, and the cross indicates the failed board. (b), (d) Resultant FACTs.

Another example is given in Fig. 3.2c, where the failure f_2 on board D even disables its sensor and thus doesn't report any alarm to the NMS. Nonetheless, it notifies its neighboring boards A, B, and C via the alarm types a_2, a_4, and a_5, respectively. f_1 causes all boards along the affected lightpath to raise alarms. On the contrary, f_2 triggers alarms a_1 and a_3 on the terminal boards G and I but bypasses the intermediate boards E, F, and H.

Clearly, both two failure scenarios f_1 and f_2 have generated the same alarm pattern, where they deploy the same network topology but different traffic distributions. Note that the same alarm name indicates the identical alarm type. Also note that whether an alarm type serves as the root alarm or the non-root alarm, as well as its correlation with other alarms, depends on both the failure type and the network state at the given moment. For instance, in Fig. 3.2a, both a_3 and a_4 are non-root alarms, with a_3 triggering a_4. Conversely, in Fig. 3.2c, a_4 is the root alarm and it activates a_3.

In the two examples above, the failure-alarm propagation resulting from a single failed board is depicted by a failure-alarm correlation tree (FACT), as shown in Fig. 3.2b and Fig. 3.2d. The aim is to identify dependent alarms to accurately pinpoint the failed device boards in the OTN optical layers. Extensive research efforts have been dedicated to developing effective alarm correlation techniques. Many of these studies focus on the relationships among alarms [59, 54, 32, 81, 8, 19, 42, 69, 84, 66, 82]. Others concentrate on the relationships between potential failed nodes/links and alarms [50, 75, 77, 46, 47, 45, 22, 35, 85, 76, 3, 36]. These studies, however, are often set in a static network environment, which may not provide sufficient generality for versatile network environments, especially those undergoing changes in network topology and traffic distribution.

Motivated by its significance and stringent demands, this chapter introduces a novel approach to realize FACT-FL, namely FACT-FL-Chain. We firstly assume that the state of each board can change at most once at the beginning of an *observation window* (OWnd), i.e., either remaining normal or transitioning from normal to failed. We then define the k^{th} *order Failure-Alarm Correlation Chain* (k-FACC) as the representation of k alarms, where each alarm triggers exactly one other due to a common failed board. Furthermore, during the OWnd, a failed board can be correlated with the collected alarms using one or multiple k-FACCs that jointly constitute an FACT. Each board/alarm incorporates a number of attributes including its occurrence time, alarm type, board type, and location.

By treating each board/alarm as a vertex, the chain-like correlation between a board and k alarms is essentially quantified by the likelihood of forming a k-FACC. To determine whether a board and k alarms can constitute a k-FACC, we employ a machine learning approach that takes into account the network topology, traffic distribution, and

alarms collected within the given OWnd. The trained binary classifiers are designed to be generalizable across different network environments with various topologies and traffic distributions, provided they adhere to the same failure-alarm propagation rules that characterize the failure-alarm propagation behavior, where those rules incorporate a particular set of failure types, board types, and alarm types. With all identified k-FACCs in place, *FACT formation* can be exclusively achieved by solving an integer linear programing (ILP) problem, where each FACT features a failed board as the tree root linked to one or multiple alarms. The objective of FACT formation is to encompass all alarms using the fewest FACTs, where each FACT captures the complete failure-alarm propagation process resulting from a specific failed board.

The contributions of this chapter are summarized as follows.

- Investigate a novel failure localization approach, namely FACT-FL-Chain, which relies on the machine learning-based binary classifiers and FACT formation modeling approach.
- Leverage the novel binary classifiers for assessing the likelihood of establishing the k-FACCs. These classifiers offer optimal generality and adaptability to versatile network environments by characterizing both boards and alarms.
- Formulate the FACT formation process as an ILP problem to yield the most optimal FACTs, ensuring that both failure localization and alarm correlation are achieved.
- Conduct extensive case studies for both single-board failure and regional failure scenarios to validate the proposed FACT-FL-Chain. This demonstrates its ability to achieve precise failure localization for optical layer device boards in OTN.

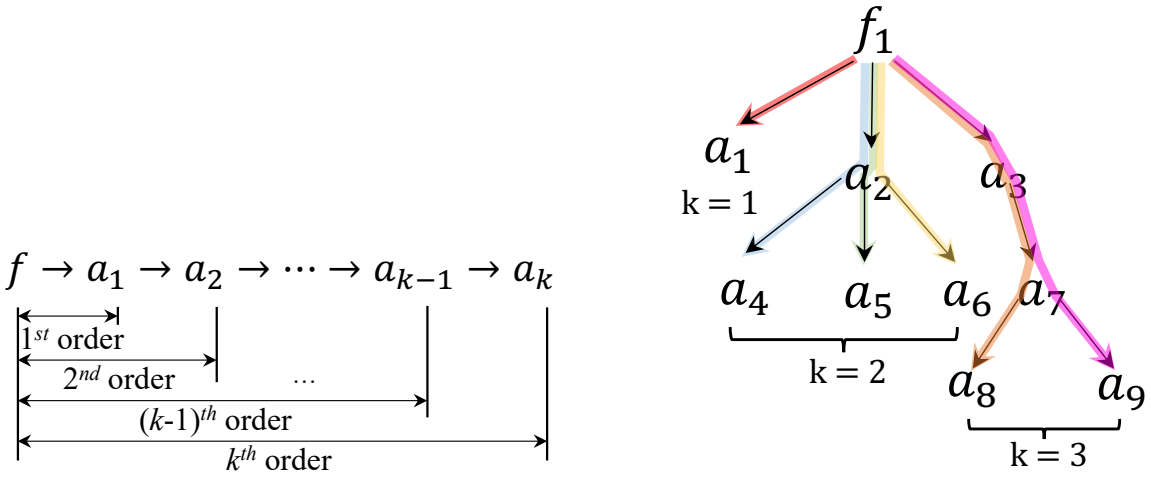
The remainder of this chapter is organized as follows: Section 3.2 details the system model. The proposed FACT-FL-Chain approach is described in Section 3.3. Section 3.4 discusses the case study setup and results. Lastly, Section 3.5 provides the concluding remarks.

3.2 System Model

Two types of instances are defined in this study. One is an *alarm instance* denoted as a 5-tuple $a = (t, l, b, m, r)$, where t is the occurrence time, l and b are the ID and type of the board that reports this alarm, respectively. m is the alarm type and r is the layer

information that can be OTS, OMS, or OCH. Another is a *board instance* that is defined by a 2-tuple $f = (l, b)$, where l and b are the board ID and type. Note that the definitions of alarm instance and board instance are consistent with the definitions in Chapter 2.2.

The instance correlation is evaluated among one board instance and k alarm instances. As illustrated in Fig. 3.3a, a correlation chain of one board instance and k alarm instances can be generally expressed as a k-FACC due to the fact that a failed board f successively triggers k alarms. Furthermore, as shown in Fig. 3.3b, an FACT can be deemed as a suite of k-FACCs that have different order values of k and result from a common failed board.



(a) (b)

Figure 3.3: (a) Illustration of a k-FACC. (b) An FACT is the ensemble of k-FACCs with various order values, where $k \in \{1, 2, 3\}$ are considered in this example.

3.3 Proposed FACT-FL-Chain Approach

Fig. 3.4 demonstrates the flowchart of the proposed FACT-FL-Chain approach. Given the raw dataset of historical observations, for all k in the range of $\{1, \cdots, K\}$, we firstly extract some features from each k-FACC, as shown in (i) of Fig. 3.4 and elaborated in Section 3.3.1. Then the attained dataset is utilized to train a binary classifier for recognizing k-FACCs, as depicted in (ii) of Fig. 3.4 and detailed in Section 3.3.2. With the learned binary classifiers, the k-FACCs derived from all possible order values of k can be leveraged for the FACT

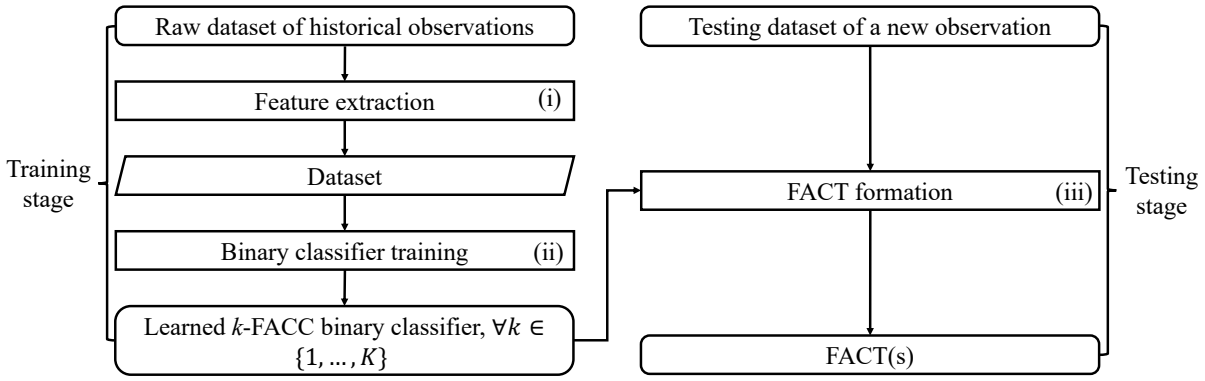


Figure 3.4: Flowchart of the proposed FACT-FL-Chain approach.

formation procedure, which is displayed in (iii) of Fig. 3.4 and explained in Section 3.3.3. Ultimately, for the given testing dataset of a new observation, one or multiple FACTs shall be gained as the output of FACT-FL-Chain.

In addition, we have summarized all notations in Table 3.1 for easier reference in the coming subsection.

Table 3.1: Summary of symbols and their definitions

Symbol	Definition
k	Order value of k-FACC
T_i	Observation window
G_{T_i}	Network topology in T_i
L_{T_i}	Traffic distribution in T_i
A_{T_i}	Alarm set in T_i
F_{T_i}	Board set in T_i
$D^k_{T_i}$	Set of k-FACC candidates in T_i
$U^k_{T_i}$	Set of ground-truth k-FACCs in T_i
D^k	Training dataset
U^k	Set of ground-truth k-FACCs

3.3.1 Feature Extraction

The raw dataset is a set of data from historical operations, each collected in a set of OWnds denoted as $\mathcal{T} = \{T_1, \cdots, T_i, \cdots, T_I\}$, $\forall i \in \{1, \cdots, I\}$. During T_i we observe the network topology G_{T_i}, traffic distribution L_{T_i}, alarm set A_{T_i}, failure event, and reference FACT(s). Here, G_{T_i} represents the connectivity of all device boards. L_{T_i} is depicted by a set of lightpaths each traversing certain device boards in G_{T_i}. $A_{T_i} = \{a_1, \cdots, a_{N_{T_i}}\}$ incorporates N_{T_i} alarm instances that have been sorted in ascending order according to their occurrence time. The board set, defined as $F_{T_i} = \{f_1, \cdots, f_{M_{T_i}}\}$, is obtained by considering all device boards in G_{T_i}. Note that the reference FACT(s) is inferred according to the failure event and A_{T_i}. Given F_{T_i} and A_{T_i}, the set of k-FACC candidates denoted as $D^k_{T_i}$ can be acquired by enumerating all entries from F_{T_i} and A_{T_i}. The set of ground-truth k-FACCs denoted as $U^k_{T_i}$ is abstracted from the reference FACT(s) in T_i to label each k-FACC candidate in $D^k_{T_i}$.

To characterize each k-FACC candidate, we create a feature space $\mathcal{H}$ with dimensions including the time, network topology, traffic distribution, and board/alarm attributes, as given by:

- For each board/alarm instance associated with a k-FACC candidate, we create a dictionary with an integer value index and it considers 15 board types, 25 alarm types, and 4 layers.
- For every pair of adjacent instances within a k-FACC candidate, we extract the following features:
  - **time gap**: the time gap between the occurrences of two instances;
  - **physical distance**: the length of the shortest path from the instance that occurred earlier to the other one in G_{T_i};
  - **hop distance**: the number of OMSs taken by the shortest path in terms of hop count;
  - **lightpath label**: a binary value taking 0 or 1 indicates whether these two instances are traversed by a common lightpath.

Mapping each k-FACC candidate from $D^k_{T_i}$ to $\mathcal{H}$ leads to a transformed dataset $\mathcal{H}(D^k_{T_i})$. Further, $\forall i \in \{1, \cdots, I\}$, $D^k = \bigcup_{i=1}^{I} \mathcal{H}(D^k_{T_i})$, $U^k = \bigcup_{i=1}^{I} U^k_{T_i}$ can be obtained as the dataset for the subsequent binary classifier training and labeling each k-FACC candidate, respectively.

Note that for each k-FACC candidate in the dataset D^k, it's labeled as 1 if it appears in U^k and 0 otherwise.

Besides, for the testing dataset observed in a new network environment that adopts the same failure-alarm propagation rules, all numerical features including the time gap, physical distance, and hop distance can be normalized by min-max scaling.

3.3.2 k-FACC Binary Classifier

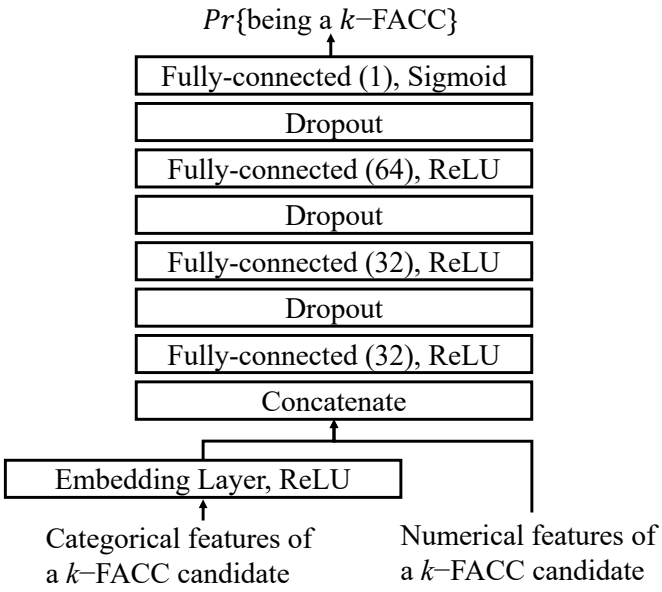


Figure 3.5: Architecture of k-FACC binary classifier.

A binary classifier is formulated to evaluate the likelihood of being a k-FACC. We employ a fully connected neural network (FCNN) whose architecture is illustrated in Fig. 3.5, where the dimension of each layer is annotated in parentheses. The features of a k-FACC candidate consist of the numerical features including the time gap, physical distance, and hop distance as well as the categorical features comprising the lightpath label and board/alarm attribute. To jointly consider these two types of input features, the categorical features are mapped onto a continuous space via an embedding layer, and its output is concatenated with the numerical features and fed into FCNN. Finally, FCNN outputs the probability of forming a k-FACC. Note that those simpler/low complexity ML models (i.e., decision tree, random forests, support vector machine, naive bayes, and k-nearest

neighbors, et al.) may output the label of each data sample without giving the likelihood, which doesn't meet the requirement of our design. Therefore, FCNN is considered in this study.

Since most board instances and alarm instances are non-correlated and cannot establish a k-FACC, the dataset D^k is imbalanced, where the number of samples from the majority class (0) is way more than that of samples from the minority class (1). Therefore, we adopt the resampling approach to build a balanced dataset through oversampling the samples from the minority class by random duplication [49] [31], i.e., the samples from the minority class can be randomly chosen and added to the new "more balanced" dataset multiple times. Note that this oversampling process may increase the likelihood of overfitting the model since it makes exact copies of the samples in the minority class. Also, this process was performed after the digitization of the categorical features and solely determined by the sample labels. However, the experimental results shown in Fig. 3.8 demonstrate that all trained k-FACC classifiers fit well on the validation set in terms of accuracy and loss. Moreover, random undersampling is an alternative approach that deletes samples in the majority class, whereas it may be more suitable for those imbalance datasets with a sufficient number of samples in the minority class and thus it is inappropriate for our scenario.

3.3.3 FACT Formation

Within each OWnd P, the testing dataset of a new observation whose structure is similar to the raw dataset is provided. It incorporates the network topology G_P, traffic distribution L_P, alarm set A_P, and the corresponding board set F_P. The proposed FACT formation algorithm aims to construct the fewest FACTs within each P while covering all alarm instances in A_P, with each FACT rooted by a failed board instance and some alarm instances as the leaves.

In the proposed FACT-FL-Chain approach, we assume that once a board instance fails at the beginning of P, it remains failed until the end of P. Further, all alarm instances in A_P are as a result of the failure event at the starting moment of P that hits one or a number of board instances.

Fig. 3.6 depicts the proposed FACT formation algorithm. Given the testing dataset during P and pre-trained binary classifiers for identifying k-FACCs, $\forall k \in \{1, \cdots, K\}$, a weighted directed acyclic multigraph $\mathcal{G}_P$ is constructed for provisioning the search space of FACT(s), denoted as $\mathcal{G}_P = (V_P, C_P, W(C_P))$. $V_P = F_P \bigcup A_P$ constitutes the vertex set, encompassing all board instances and alarm instances. C_P is the set of k-FACCs

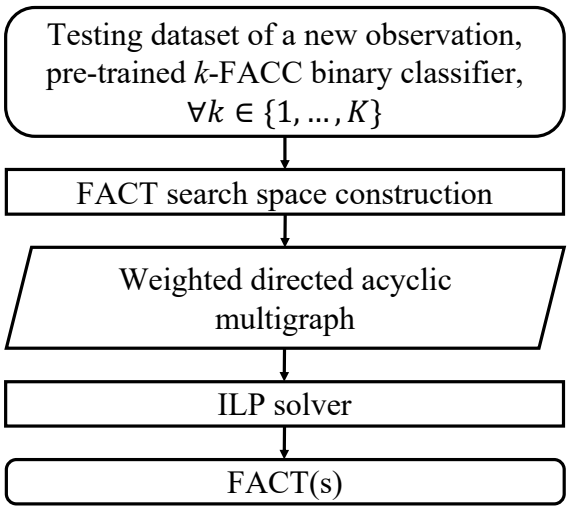


Figure 3.6: Flowchart of the proposed FACT formation algorithm.

recognized by those classifiers and each k-FACC is portrayed by a directed path, denoted as $c = \{f_c\} \bigcup A_c$, where $\forall c \in C_P, f_c \in F_P, A_c \subseteq A_P$. Also, $W(C_P)$ is the set of non-negative weights of all k-FACCs in C_P, where $\forall w_c \in W(C_P)$ indicates the weight/cost of k-FACC c and it's defined in (3.1):

$$w_c = 1 - Pr(f_c \text{ is failed}) \cdot Pr(c \text{ is a } k\text{-FACC}), \tag{3.1}$$

where $Pr(f_c$ is failed) is the probability that the board instance f_c becomes failed. It can be determined by the probability density function (PDF) of the corresponding board's time-to-failure, where the PDF can be estimated according to the historical failure events taking place at this board. $Pr(c$ is a k-FACC) is the probability output from the trained k-FACC binary classifier. Note that to reduce the computational complexity of the FACT formation algorithm, only the k-FACCs whose probability output from the classifier is greater than 0.5 are involved in creating $\mathcal{G}_P$. Also, note that a board instance that doesn't form k-FACC with any alarm instance will not join the FACT formation process and won't be considered as failed.

Furthermore, the problem of abstracting the most likely FACTs $\mathcal{G}_P^{opt}$ from $\mathcal{G}_P$ can be formulated as an ILP problem given as follows:

$$\text{minimize} \quad \sum_{c \in C_P} w_c x_c + \sum_{u \in F_P} (1 - Pr\{u \text{ fails}\}) \cdot y_u \tag{3.2a}$$

$$\text{subject to} \quad x_c \leq y_u, \quad \forall u \in F_P, \forall c \in \delta^+(u), \tag{3.2b}$$

$$\sum_{c \in \delta^-(v)} x_c \geq 1, \forall v \in A_P, \tag{3.2c}$$

$$\sum_{c \in \delta^-(v)'} x_c + \frac{\sum_{c \in \delta^-(v) \setminus \delta^-(v)'} x_c}{|\delta^-(v) \setminus \delta^-(v)'|} \leq 1, \forall v \in A_P, \tag{3.2d}$$

$$x_i + x_j \leq 1, \forall i \in C_l \subseteq C_v^u, \forall j \in C_v^u \setminus C_l,$$
$$\forall l \in \{1, \cdots, n_v^u\}, \forall u \in F_P, \forall v \in A_P, \tag{3.2e}$$

$$x_c \in \{0, 1\}, \forall c \in C_P, \tag{3.2f}$$

$$y_u \in \{0, 1\}, \forall u \in F_P. \tag{3.2g}$$

$\forall c \in C_P, \forall u \in F_P$, two binary variables x_c, y_u are defined, where x_c takes 1 if the k-FACC c is chosen by $\mathcal{G}_P^{opt}$ and 0 otherwise; while y_u takes 1 if the board vertex u is selected as a tree root in $\mathcal{G}_P^{opt}$ and 0 otherwise. The objective function (3.2a) aims to find $\mathcal{G}_P^{opt}$ that minimizes the total cost of selected k-FACCs and board instances. Constraint (3.2b) implies that for each board vertex u, if any item from the set of k-FACCs outgoing from u, defined as $\delta^+(u)$, is chosen, then u must be selected. Constraint (3.2c) suggests that for each alarm vertex v, at least one item from the set of k-FACCs incoming to v, namely $\delta^-(v)$, must be opted to ensure all alarm instances are visited by $\mathcal{G}_P^{opt}$. Constraint (3.2d) aims to guarantee that the selected k-FACCs won't form a closed loop, which indicates that for each v, one can either choose exactly one item from the set of k-FACCs terminating at v, referred to as $\delta^-(v)$, or choose one or multiple items from the set of k-FACCs bypassing v, represented by $\delta^-(v) \setminus \delta^-(v)'$. An illustrative example is given in Fig. 3.7, where f_1 indicates the board vertex, $\{e_1, e_2, e_3, e_4\}$ is the set of k-FACCs, and $\{a_1, \ldots, a_8\}$ is the set of alarm vertices. For a_4, we can either select one item from $\{e_1, e_4\}$ or select one or multiple items from $\{e_2, e_3\}$. Constraint (3.2e) ensures that for each v and u, the selected k-FACCs, which

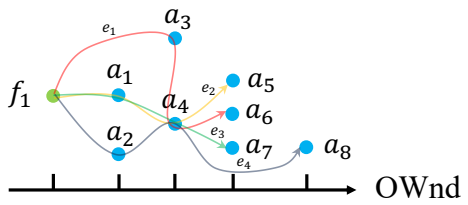


Figure 3.7: Illustrative explanation of ILP constraint (3.2d).

start from u and bypass v, must share a common trajectory before reaching v. Here, C_v^u is the set of k-FACCs that originate from u and bypass v, which can be further divided into

n_v^u subsets based on various trajectories from u to v. Note that (3.2d) and (3.2e) aim to ensure that the in-degree of each alarm instance must be one such that the $\mathcal{G}_P^{opt}$ satisfies the property of being a set of trees.

By solving the above ILP, we can obtain the anticipated FACT(s) $\mathcal{G}_P^{opt}$, which realizes both failure localization and alarm correlation.

3.4 Case Studies

Extensive case studies are conducted to verify the effectiveness of the proposed FACT-FL-Chain in the OTN optical layers and compare it with a number of counterparts. An OTN simulator [48][44] is firstly developed to generate ground-truth alarms and resultant FACTs according to the given failure event, failure-alarm propagation rule database, as well as the network topology and traffic distribution during an OWnd. Currently, the rule database contains 38 entries in the form of *One2Many*, where 16 failure types, 16 board types, and 25 alarm types are taken into account. Without loss of generality, each failure event independently hits a certain number of boards and affects their traversing optical flows (i.e., OTS, OMS, and/or OCH). Meanwhile, we assume that the failure-alarm propagation process always succeeds even if there is any malfunctioned board along the notification signal propagation path. On the other hand, we assume that each board instance is even likely to fail and thus ignore the cost of choosing board instances by setting $Pr\{u \text{ fails}\}$ as 1 in (3.2a).

The goal of the case studies is to validate the generality and migratability of FACT-FL-Chain. For this, we apply the k-FACC binary classifiers trained by the raw dataset to the testing datasets obtained from the OTN environments with different failures, network topologies, traffic distributions, and noise alarms. The state-of-the-art counterparts considered in the case studies include BP [50], LSTM [50], convolutional neural network (CNN) [77], GNN [46, 47, 45], BERT [35], and FACT-FL-Heuristic [37][39]. Additionally, we evaluate the performance of FACT-FL-Chain under various maximum order values of k denoted as K, where 1, 2, and 3 are considered due to the fact that the OTN optical domain adopts a 3-sublayer structure. Note that the value of K equals the number of layers considered by the network protocol.

3.4.1 Case Study Setup

Raw Dataset

The raw dataset is generated from a default network state that consists of 8 nodes and 304 boards, where it deploys 20 lightpaths each averagely traversing 14 boards, and the board-level average degree is 2.31. We collect 3730 alarms in total from the default network state via 304 1-minute OWnds, where those OWnds are launched by the single-board failure events that independently hit each board in the network topology.

Datasets

The raw dataset leads to the datasets of size 146247, 28316, and 150249 for training the 1-FACC, 2-FACC, and 3-FACC classifiers, where their imbalance ratios are 0.09%, 1.04%, and 0.55%, respectively. Note that the dataset size is determined by the number of k-FACC candidates, where those candidates are enumerated by the failed boards and the received alarms involved in all reference FACTs. Also, each reference FACT has a unique tree structure, resulting in various numbers of k-FACC candidates. Each dataset is set at 64%, 16%, and 20% split for training, validation, and test sets. The binary cross-entropy loss function is optimized with Adam at a learning rate of 0.001. We set the batch size and number of epochs to 450 and 100. Also, we monitor the value of the area under the curve on the validation set in each epoch and use early stopping [7] to reduce overfitting.

Machine Learning Models of the Counterparts

The network architectures of the counterparts using the raw dataset are briefly described as follows. For BP and LSTM, two networks of dimension $128 \times 64 \times 32 \times 304$ and $128 \times 64 \times 304$ are constructed, where the input and output network dimensions are determined by the number of all possible alarming boards and the total number of boards in the given network topology. For GNN, the alarm knowledge graph is constructed according to our rule database, leading to 41 entries. CNN model consists of 24 input layer units and 3 hidden layers whose number of neurons are 256, 128, and 32, where the kernel size in each hidden layer is 3×3. BERT encodes the alarm context with a 768-dimension vector, where it sets the length of each alarm transaction to 3 according to the maximum order value of k. Note that for BP, LSTM, and CNN model, those hyperparameters taken from those references are reasonable considering their input and output network dimensions.

Testing Datasets

Testing datasets are produced as below. To evaluate the benchmark performance of each scheme, the single-board failure dataset is prepared that contains 10 failed boards randomly chosen from the default network state.

Furthermore, we generate two regional failure datasets to assess the generality and migratability of each scheme. One dataset incorporates the regional failures from 10 new network states by changing the number of lightpaths in a consistent network topology. The network topology consists of 10 nodes, 2380 boards, and its board-level average degree is 2.06. The number of lightpaths varies from 10 to 100 at a constant interval of 10, and each lightpath averagely traverses 13 boards. Another dataset includes the regional failures from 10 new network states each deploying 40 lightpaths. These states result from diverse topological sizes, where the number of nodes, the number of boards, the board-level average degree, and the average number of boards traversed by each lightpath range from 11 to 20, 461 to 700, 2.31 to 2.38, and 8 to 18, respectively. *Note that each regional failure is initiated by randomly selecting one board from the network topology, which is further expanded by arbitrarily adding one board at a time, ensuring that the chosen boards are connected.* The number of failed boards in each regional failure varies from 1 to 10, and the average degree of each regional failure is 1.25.

Finally, the single-board failure dataset is created to test the anti-noise capability of FACT-FL-Chain, which contains 10 failed boards, and different ratios of noise alarms are introduced on top of the true alarms due to each failed board. Note that those noise alarms are randomly selected from diverse OTN sublayers.

Performance Metrics

$$F\text{-}measure(R) = 2 \cdot \frac{Precision(R) \cdot Recall(R)}{Precision(R) + Recall(R)}, \tag{3.3}$$

$$Precision(R) = \frac{\#\text{ of correctly inferred root alarms}}{\text{total }\#\text{ of inferred root alarms}},$$

$$Recall(R) = \frac{\#\text{ of correctly inferred root alarms}}{\text{total }\#\text{ of root alarms}},$$

$$F\text{-}measure(F) = 2 \cdot \frac{Precision(F) \cdot Recall(F)}{Precision(F) + Recall(F)}, \tag{3.4}$$

$$Precision(F) = \frac{\#\text{ of correctly inferred failed boards}}{\text{total }\#\text{ of inferred failed boards}},$$

$$Recall(F) = \frac{\#\text{ of correctly inferred failed boards}}{\text{total }\#\text{ of failed boards}},$$

$$F\text{-}measure(T) = 2 \cdot \frac{Precision(T) \cdot Recall(T)}{Precision(T) + Recall(T)}, \tag{3.5}$$

$$Precision(T) = \frac{\#\text{ of correctly inferred 1-FACCs, 2-FACCs, and 3-FACCs}}{\text{total }\#\text{ of inferred 1-FACCs, 2-FACCs, and 3-FACCs}},$$

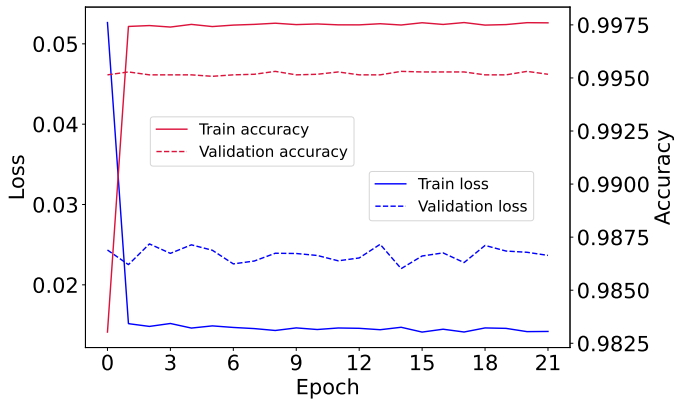
$$Recall(T) = \frac{\#\text{ of correctly inferred 1-FACCs, 2-FACCs, and 3-FACCs}}{\text{total }\#\text{ of 1-FACCs, 2-FACCs, and 3-FACCs}}.$$

The performance metrics called *F-measure(R)*, *F-measure(F)*, and *F-measure(T)* are considered for evaluating the following three outcomes from our approach in each simulated instance, namely *root alarms*, *failed boards*, and FACTs. The definitions of *F-measure(R)*, *F-measure(F)*, and *F-measure(T)* are given in (3.3), (3.4), and (3.5), where a higher value represents better performance in identifying root alarms/failed boards/FACTs. Note that *F-measure(T)* is merely applicable to FACT-FL-Chain with $K = 3$.

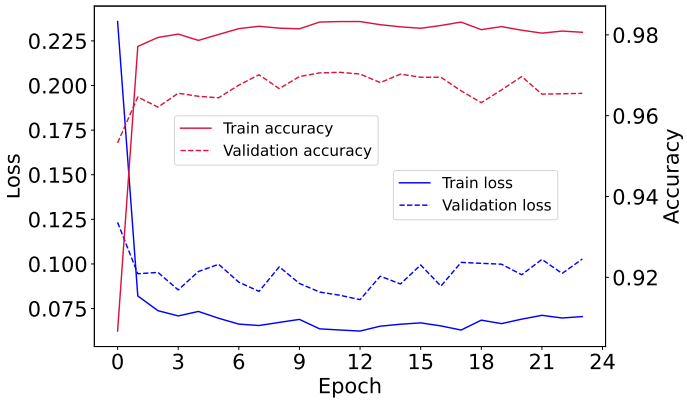
3.4.2 Results

k-FACC Binary Classifiers

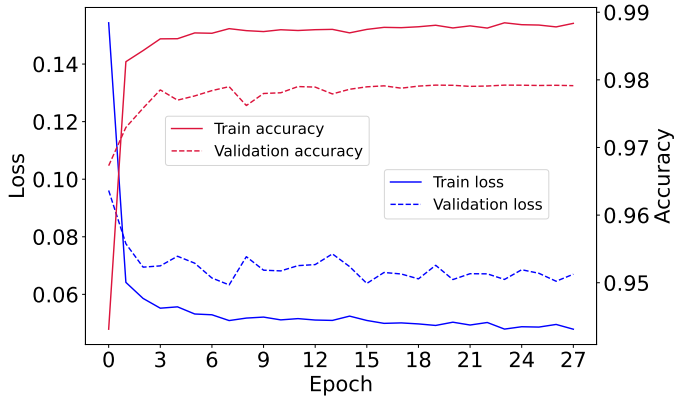
To determine the architecture of k-FACC binary classifiers, we range the number of hidden layers from 1 to 10 and the number of neurons per hidden layer from {16, 32, 64, 128, 256}. For simplicity, we fix the number of neurons in the last hidden layer at 64. We leverage the strategy of grid search with cross-validation for finding the optimal ones, where the most



(a) 1-FACC classifier



(b) 2-FACC classifier



(c) 3-FACC classifier

Figure 3.8: Training curves of k-FACC binary classifiers.

suitable k-FACC binary classifiers consist of 3 hidden layers whose number of neurons are 32, 32, and 64, respectively.

The training curves of 1-FACC/2-FACC/3-FACC binary classifiers are demonstrated in Fig. 3.8. The loss and accuracy of corresponding classifiers on the training set and validation set are summarized in Table 3.2, which show that the obtained classifiers have well converged.

Table 3.2: Classifier performance on the training/validation set.

Classifier	# of trainable parameters	# of training epochs	Training set		Validation set	
			Loss	Accuracy	Loss	Accuracy
1-FACC	5265	22	0.0142	99.76%	0.0237	99.52%
2-FACC	6129	24	0.0705	98.06%	0.1028	96.55%
3-FACC	6993	28	0.0479	98.84%	0.0670	97.91%

FACT Formation

Table 3.3: Average performance comparison on the single-board failure dataset

Method	# of trainable parameters	*F-measure(R)*	*F-measure(F)*
FACT-FL-Chain, $K = 3$	18771	1.000±0.000	1.000±0.000
FACT-FL-Chain, $K = 2$	11490	0.886±0.122	0.773±0.203
FACT-FL-Chain, $K = 1$	5265	0.783±0.217	0.750±0.250
FACT-FL-Heuristic (IC-FD)	5585	0.608±0.259	0.369±0.201
CNN	338441	0.877±0.106	N/A
BERT	2446910	0.070±0.124	
BP	44528	N/A	0.874±0.192
LSTM	69168		0.900±0.186
GNN	51999		0.531±0.225

Firstly, the experimental results of various schemes on the single-board failure dataset are summarized in Table 3.3, where the value before "±" implies the average *F-measure(R)*/*F-measure(F)* over 10 failures and the value after "±" indicates the margin of error at the confidence interval of 95%. Apparently, FACT-FL-Chain with $K = 3$ achieves a significant advantage in all employed metrics while taking the fewest trainable parameters and its *F-measure(T)* reaches 0.912 ± 0.059. Compared with $K = 3$, FACT-FL-Chain with $K = 2$ suffers from severe performance degradation and it is even worse for $K = 1$. This verifies

the necessity of considering all possible chain order values, given that the chain order values involved in a failure event are co-determined by the failure type, network topology, and traffic distribution.

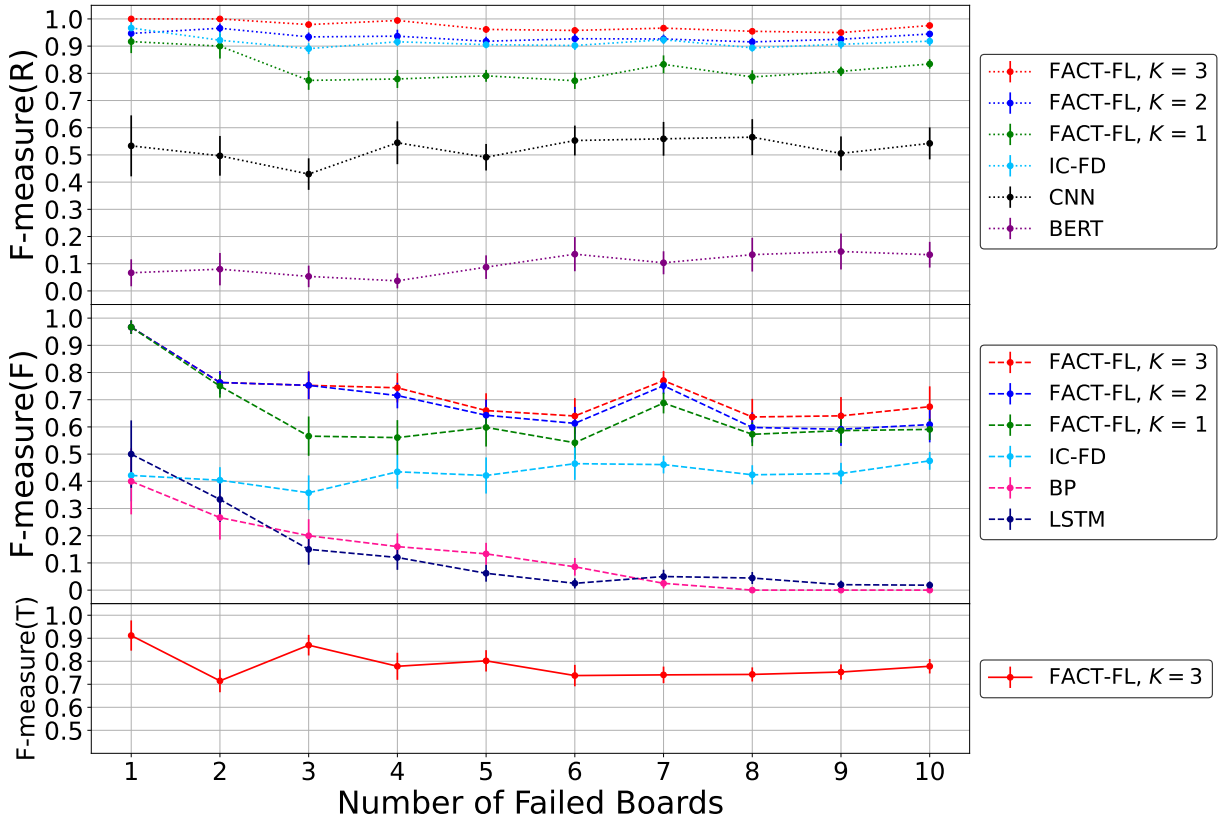


Figure 3.9: Average performance result on the regional failure dataset under various traffic distributions.

Furthermore, the average performance results of different schemes on two regional failure datasets are shown in Fig. 3.9 and Fig. 3.10. As for recognizing root alarms, FACT-FL-Chain with $K = 3$ outperforms the counterparts under all considered number of failed boards, while FACT-FL-Chain with $K = 2$ behaves similar to FACT-FL-Heuristic but much better than FACT-FL-Chain with $K = 1$. However, CNN and BERT yield significantly degraded performance since they completely ignore the traffic distribution and spatial relation among those alarms. Moreover, as for identifying failed boards, FACT-FL-Chain with $K = 3$ achieves the best performance among all its counterparts and the performance of FACT-FL-Chain obtains slight performance degradation when choosing

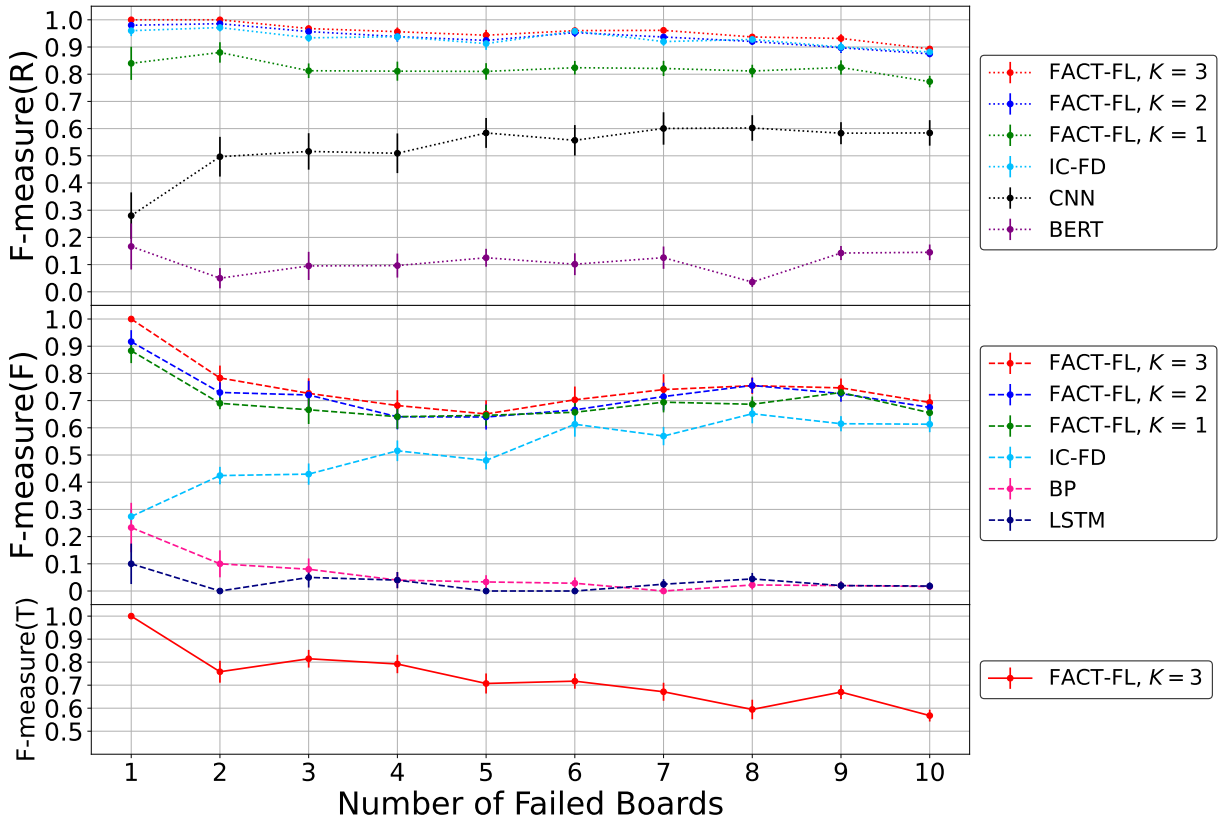


Figure 3.10: Average performance result on the regional failure dataset under various network topologies.

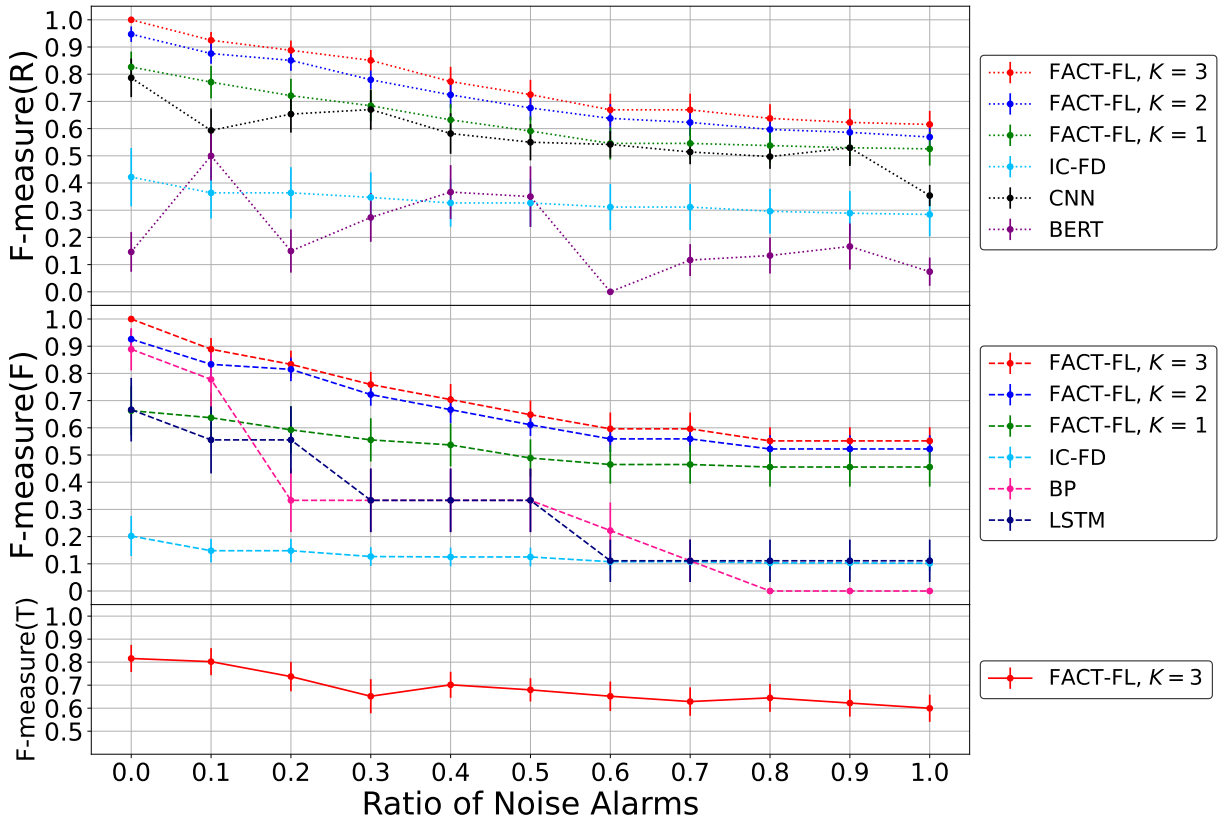


Figure 3.11: Average performance result on the single-board failure dataset under various ratios of noise alarms.

the smaller K, whereas it is still better than FACT-FL-Heuristic. Nevertheless, BP and LSTM show their serious performance decline with the increase in the number of failed boards, which suggests that they are likely to miss more real failed boards while abundant alarm attributes aren't considered.

In addition, the average performance of different schemes on the single-board failure dataset with noise alarms is displayed in Fig. 3.11. FACT-FL-Chain with a larger K demonstrates a good capability in overcoming the noise alarms thanks to its more thorough consideration of the correlation among the received alarms, while all other counterparts suffer crucial performance degradation in detecting root alarms/failed boards as the ratio of noise alarms continues to increase.

As for the ILP solver, we adopt the branch-and-price method [4] that works for ILP problems with a huge number of variables, where it is implemented in the high-performance optimization software (HiGHS) [23] that is designed for large scale sparse linear optimization models.

Note that since the location information of failed board/alarm varies with different network environments, all models taken by the counterparts require retraining when there is any change with the network topology/traffic distribution, whereas FACT-FL-Chain and FACT-FL-Heuristic only need to be trained once with the alarms collected from the default network state and thus achieve the best migratability among all the counterparts.

Complexity Analysis

We assume that during an OWnd P, the board set $F_P = \{f_1, \cdots, f_{M_P}\}$ and alarm set $A_P = \{a_1, \cdots, a_{N_P}\}$ are given. A_P can be divided into R_P subsets according to various occurrence timestamps, denoted as $A_P = \bigcup_{i=1}^{R_P} A_i$ and $N_P = \sum_{i=1}^{R_P} N_i$, where A_i and N_i represent the alarm set with the i^{th} occurrence timestamp and the number of alarms in A_i, respectively. The number of examined 1-FACC, 2-FACC, and 3-FACC candidates are $M_P \cdot N_P$, $M_P \cdot [\binom{N_P}{2} - \sum_{i=1}^{R_P} \binom{N_i}{2}]$, and $M_P \cdot \{\binom{N_P}{3} - \sum_{i=1}^{R_P} \binom{N_i}{3} - \sum_{i=1}^{R_P-1} [\binom{N_i}{2} \cdot \sum_{j=i+1}^{R_P} N_j + N_i \cdot \sum_{j=i+1}^{R_P} \binom{N_j}{2}]\}$, leading to the computation complexity of $\mathcal{O}(M_P \cdot N_P)$, $\mathcal{O}(M_P \cdot N_P^2)$, and $\mathcal{O}(M_P \cdot N_P^3)$. This suggests that the FACT-FL-Chain with a larger order value of K could incur higher computation complexity although it has achieved better performance than the FACT-FL-Chain with lower K in variable network environments. Note that this complexity could be further reduced by discarding the candidates whose categorical features don't conform to the existing k-FACCs in the dataset.

3.5 Conclusion

This chapter introduced FACT-FL-Chain, a novel failure localization approach to realize the FACT-FL framework. The presented FACT-FL-Chain features a suite of novel modeling approaches. Firstly, we have jointly considered the boards and alarms in the correlation process that localizes the failed boards while identifying the root alarms. Secondly, we have trained a set of FCNN-based binary classifiers that capture chain-like failure-alarm correlation patterns to accommodate versatile network environments. Thirdly, we have proposed the concept of FACT that effectively models the correlation among failed boards and alarms, and the FACT formation process is formulated as an elegant ILP that figures out a high-quality result.

Extensive case studies are conducted in different network environmental variations to compare the performance of the proposed FACT-FL-Chain with other schemes. We observe that FACT-FL-Chain has gained superb and stable performance in identifying failed boards and root alarms, which verifies its sufficient generality and migratability in versatile network environments.

Chapter 4

FACT-FL-GNN

This chapter proposes another novel approach to implementing the FACT-FL framework, dubbed FACT-FL-GNN. Foremost, a collection of functional graphs (FGs) is garnered by iteratively tagging each board in the network topology, serving as the ground of the proposed method. To evaluate the edge weights of potential FACTs induced by FGs, a graph neural network (GNN) with the graph transformer operator is employed as an edge classifier, which characterizes each vertex/edge from diverse dimensions including the time, traffic distribution, network topology, and board/alarm attributes. Subsequently, we frame an integer linear programing (ILP) problem to construct the most likely FACT(s). Extensive case studies are conducted to showcase FACT-FL-GNN's advantage over its counterparts in terms of the metrics assessing the identified failed boards/root alarms. We also delve into its performance in volatile environmental variations such as diverse failure scenarios, network topologies, traffic distributions, and noise alarms.

4.1 Introduction

Internet backbones have undergone explosive growth in bandwidth requirement and geographical coverage, aiming at adapting to the multi-service and multi-tenant heterogeneous network environment. Optical transport networks (OTNs) are formalized by ITU-T as a set of telecommunication protocols [29], allowing for multiplexing different services onto a single high-capacity wavelength to underpin the rapid advancement of contemporary Internet backbones. The OTN optical layers encompass the optical transport section (OTS), optical multiplex section (OMS), and optical channel (OCH) [30]. Specifically, each OCH (i.e. lightpath) traverses various types of device boards connected in series, such as the

fiber segment, optical amplifier (OA), fiber interface unit (FIU), optical multiplexer (OM), optical demultiplexer (OD), optical transponder unit (OTU), etc. [28] The OTN control plane harnesses a comprehensive set of rigid alarming mechanisms at each board for monitoring system performance. As a result, each board is capable of generating alarms in response to any failure event perceived by its sensor. For example, if an OD board detects a failure in its upstream fiber segment, it will notify the network management system (NMS) via an alarm. Alternatively, an OTN board will report an alarm to the NMS if any exception degrades the received lightpath's quality of service.

Overall, a failure event may interfere with one or multiple boards, resulting in tearing down numerous lightpaths by accident. Concurrently, other boards along the affected lightpaths may sense these disruptions as well. Additionally, a board might emit an alarm and send it to the NMS not only due to an identified failure event but also in response to the notification alarm stemming from another remote board. Accordingly, a failure event inevitably gives rise to an alarm flood, considerably boosting the complexity of alarm analysis and failure localization, particularly in the network domains with a myriad of boards such as the OTN.

Alarm correlation is universally acknowledged as an effective approach to discerning the dependencies among alarms. The objective is to remove those dependent alarms as many as possible, thereby substantially decreasing the complexity of failure localization. An illustrative example is given in Fig. 4.1(a). The fiber segment f_B^D is taken as the tagged board, attaining the *functional graph* (FG) that incorporates all boards traversed by the functional connections pertaining to the tagged board such as OTS, OMS, and OCH. A failure that hits f_B^D is firstly detected by its downstream board B. B issues an alarm a_3 and informs f_B^D's upstream board D via a_2, where these alarms are further reported to the NMS. In addition, the impacts of f_B^D are detected by all other boards traversing the affected lightpaths, engendering a_1 and a_4. Note that fiber segments are passive devices, thereby not reporting any alarms but could be failed.

Another example is illustrated in Fig. 4.1(c). The failure on the tagged board D even malfunctions its sensor and hence doesn't raise any alarm. However, D's neighboring boards A, B, and C receive the notification alarms a_2, a_1, and a_4, respectively. B causes E to report a_3, where E passes through the interrupted lightpath. Apparently, the failures on f_B^D and D influence an identical network topology of different traffic distributions, whereas they yield the same alarm pattern. Besides, the failure type and current network state determine whether an alarm is the root alarm or non-root alarm and its correlation with other alarms. For instance, in Fig. 4.1(a), a_3 is the root alarm and it triggers a_4. Conversely, in Fig. 4.1(c), a_4 is the root alarm, yet it doesn't bring about a_3.

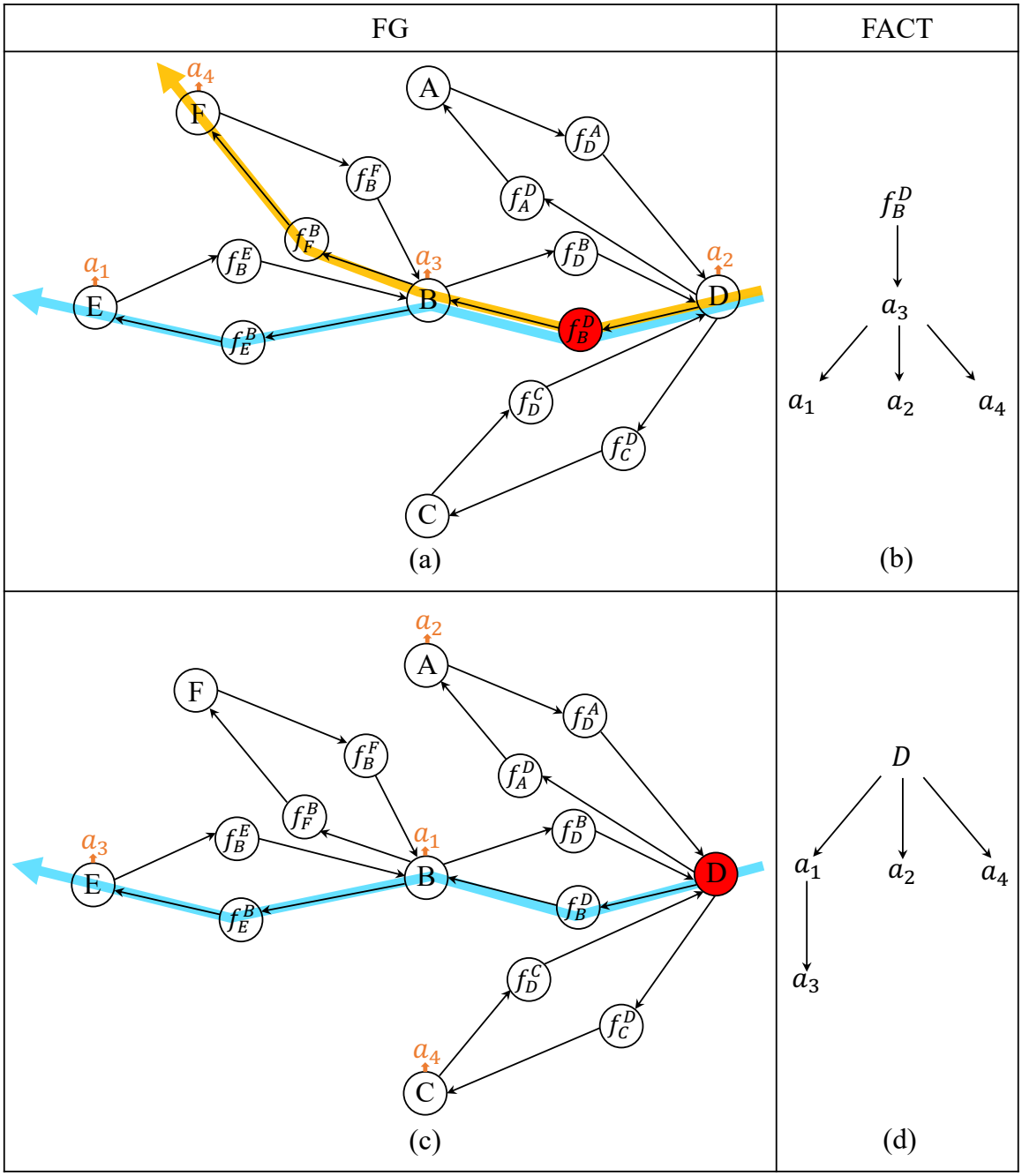


Figure 4.1: (a), (c) Illustrations of two FGs due to single-board failure events, where $A, \cdots, F, f_D^A, \cdots, f_B^F$ are device boards, bold arrows are lightpaths, red nodes are failed tagged boards, and $a_1, \cdots, a_4$ are alarm types. (b), (d) Consequential FACTs.

According to the examples above, the failure-alarm propagation phenomenon resulting from a failed tagged board in an FG is portrayed by an FACT, as displayed in Fig. 4.1(b) and Fig. 4.1(d). The goal is to recognize dependent alarms for precisely localizing failed device boards in the OTN optical layers. Extensive research achievements are directed toward proposing effective alarm correlation approaches to assist failure localization. Some of their findings concentrate on utilizing the relationships among alarms [66, 82, 77, 35, 3, 47, 46, 45, 36, 84]. Others exploit the relationships between failed nodes/links and alarms [50, 75, 85, 76, 20, 22, 70, 71, 78, 79, 34, 74]. Nevertheless, these works generally suppose that the network environment is simply static, which may not provide sufficient generality for the changeable network environment, especially for those experiencing alterations in network topology or traffic distribution.

Inspired by its importance and unique challenges, this chapter introduces a novel approach to fulfill FACT-FL, called FACT-FL-GNN. Firstly, we assume that during an *observation window* (OW) of the given network state, the status of each board can change no more than once at the beginning of this OW, i.e., either maintaining normal or transitioning from normal to failed, incurring an alarm set. Then we define an FG as a tagged board-centric graph that contains all boards functionally related to the tagged one. Based on an FG, an FACT represents the tree-shape correlation between the failed tagged board and alarms, where each board/alarm is featured by several attributes including its occurrence time, board type, and alarm type.

Given a set of FGs within an OW, FACT-FL-GNN adopts the graph neural network (GNN) as an edge classifier for weighting the edges in each potential FACT. The trained graph edge binary classifier jointly considers the network topology, traffic distribution, and received alarms. It is intended to accommodate diverse network environments, provided that they comply with the same failure-alarm propagation rules. With all weighted edges in place, *FACT formation* can be exclusively realized by tackling an integer linear programing (ILP) problem, where each FACT, connecting to one or multiple alarms, is rooted by a failed board. The purpose of FACT formation is to cover all alarms with the minimum cost.

The contributions of this chapter are summarized as follows.

- Propose a novel board-level failure localization approach, referred to as FACT-FL-GNN, which leans on a graph edge binary classifier and FACT modeling approach.
- Leverage GNN for building a novel graph edge binary classifier. This classifier offers superior adaptability and generality to versatile network environments by considering both boards and alarms.

- Formulate the FACT formation problem as an ILP to obtain the optimal FACT(s), fulfilling both failure localization and alarm correlation.
- Conduct extensive case studies on the single-board and regional failure events to verify the raised FACT-FL-GNN. The results demonstrate its capability of precisely localizing failed device boards in the OTN optical layers.

The remainder of this chapter is structured as follows. Section 4.2 dwells on the proposed FACT-FL-GNN approach. Section 4.3 describes the setup of case studies and results. Section 4.4 concludes this chapter.

4.2 Proposed FACT-FL-GNN Approach

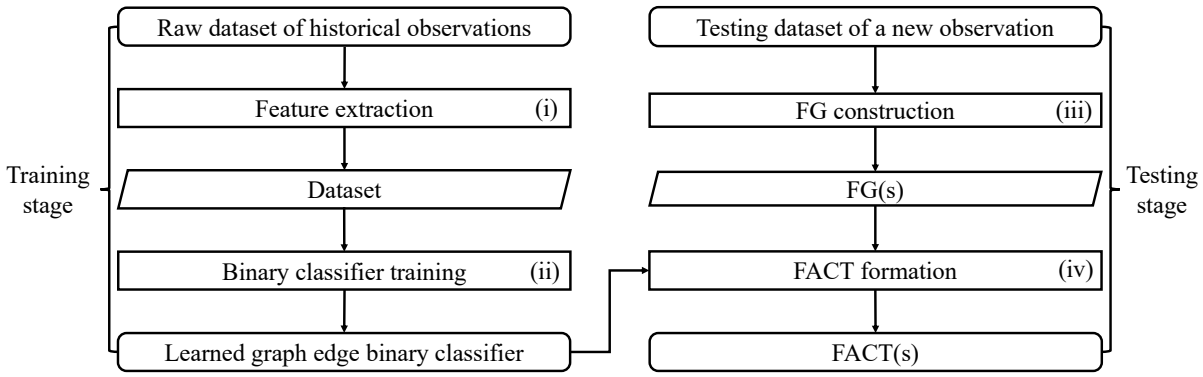


Figure 4.2: Flowchart of the proposed FACT-FL-GNN approach.

Fig. 4.2 shows the flowchart of the proposed FACT-FL-GNN approach. Given the raw dataset of historical observations, we extract the features from each FACT, as depicted in (i) of Fig. 4.2 and detailed in Section 4.2.1. Then the obtained dataset is harnessed for training a graph edge binary classifier to weight the edges in each potential FACT, as exhibited in (ii) of Fig. 4.2 and elaborated in Section 4.2.2. Further, a new observation is furnished as the testing dataset, aiming at attaining a number of FGs via FG construction, as displayed in (iii) of Fig. 4.2 and described in Section 4.2.3. The learned classifier and FG(s) are the inputs for the FACT formation procedure demonstrated in (iv) of Fig. 4.2 and explained in Section 4.2.4. FACT-FL-GNN culminates in gaining one or multiple FACTs as the expected output.

Also, we have summarized all notations used in this chapter in Table 4.1 for readability.

Table 4.1: Summary of symbols and their definitions

Symbol	Definition
O_i	Observation window
T_{O_i}	Network topology in O_i
L_{O_i}	Traffic distribution in O_i
A_{O_i}	Alarm instance set in O_i
N_{O_i}	Number of alarm instances in A_{O_i}
B_{O_i}	Board instance set in O_i
D_{O_i}	Dataset in O_i
D	Training dataset

4.2.1 Feature Extraction

The raw dataset is composed of historical observations collected from a set of OWs, denoted as $\mathcal{O} = \{O_1, \cdots, O_i, \cdots, O_I\}$, $\forall i \in \{1, \cdots, I\}$. During O_i we observe the network topology T_{O_i}, traffic distribution L_{O_i}, alarm set A_{O_i}, failure event, and FACTs. Here, $T_{O_i} = (B_{O_i}, E_{O_i})$ is an undirected graph indicating the board-level connectivity, where B_{O_i} is the vertex set of device board instances and E_{O_i} is the edge set. L_{O_i} represents a set of lightpaths each traversing several device board instances in B_{O_i}. $A_{O_i} = \{a_1, \cdots, a_{N_{O_i}}\}$ embodies N_{O_i} alarm instances which are sorted in ascending order of their occurrence time. Note that the FACTs could be inferred by the failure event and B_{O_i} with the assistance of expertise, and we assume that they are consistent with the ground-truth failure-alarm propagation rules. For each FACT, all its edges are labeled as 1s. Also, we introduce the auxiliary edges for creating the edges labeled as 0s. Some auxiliary edges are carried out by connecting the tree root board instance with each alarm instance, while others are realized by pairing any two alarm instances with a positive occurrence time gap. Eventually, these modified FACTs constitute the dataset D_{O_i}.

To portray each FACT in D_{O_i}, we create a feature space $\mathcal{F}$ with dimensions including the time, network topology, traffic distribution, and board/alarm attributes, given as below:

- For each vertex within an FACT, we consider the board type, alarm type, and layer information.

- For each edge within an FACT, we extract the following features:
  - **occurrence time gap**: the occurrence time gap between two instances;
  - **physical distance**: the length of the shortest path from the instance that happened earlier to the other one in T_{O_i};
  - **hop distance**: the number of OMSs passed by the shortest path in terms of hop count;
  - **lightpath label**: a binary value signifying whether these two instances traverse a common lightpath.

Mapping each FACT from D_{O_i} to $\mathcal{F}$ results in a transformed dataset $\mathcal{F}(D_{O_i})$. Ultimately, $\forall i \in \{1, \cdots, I\}$, $D = \bigcup_{i=1}^{I} \mathcal{F}(D_{O_i})$ can be garnered as the dataset catering to the need of training the graph edge binary classifier.

4.2.2 Graph Edge Binary Classifier

A graph edge binary classifier is designed to evaluate the edge weights for each potential FACT. We adopt a one-layer GNN whose architecture is shown in Fig. 4.3, where the dimension of the corresponding layer/embedding vector is annotated in parentheses. The input layer consists of vertex features, edge features, and FACT. The features of the vertex and edge belong to categorical and numerical ones, respectively. To jointly manipulate these two types of features, the vertex features are mapped onto a continuous space via an embedding layer. Then the vertex and edge features are individually pre-processed with 2 fully connected layers. Each FACT, which indicates the connectivity amongst these vertices, is treated as an undirected graph due to the fact that the performance of benchmark GNN models on homophilic graphs didn't benefit much from using edge direction [58]. The GNN layer employs 1 convolutional layer with the graph transformer operator [61] that supports message passing with vertex and edge features along the FACT structure, outputting the vertex embedding vectors. Consequently, a graph edge binary classifier is established to calculate the probability of an edge connecting two vertices as the edge weight. We concatenate the two vertices' embedding vectors as well as the corresponding edge embedding vector and feed it into another fully connected layer.

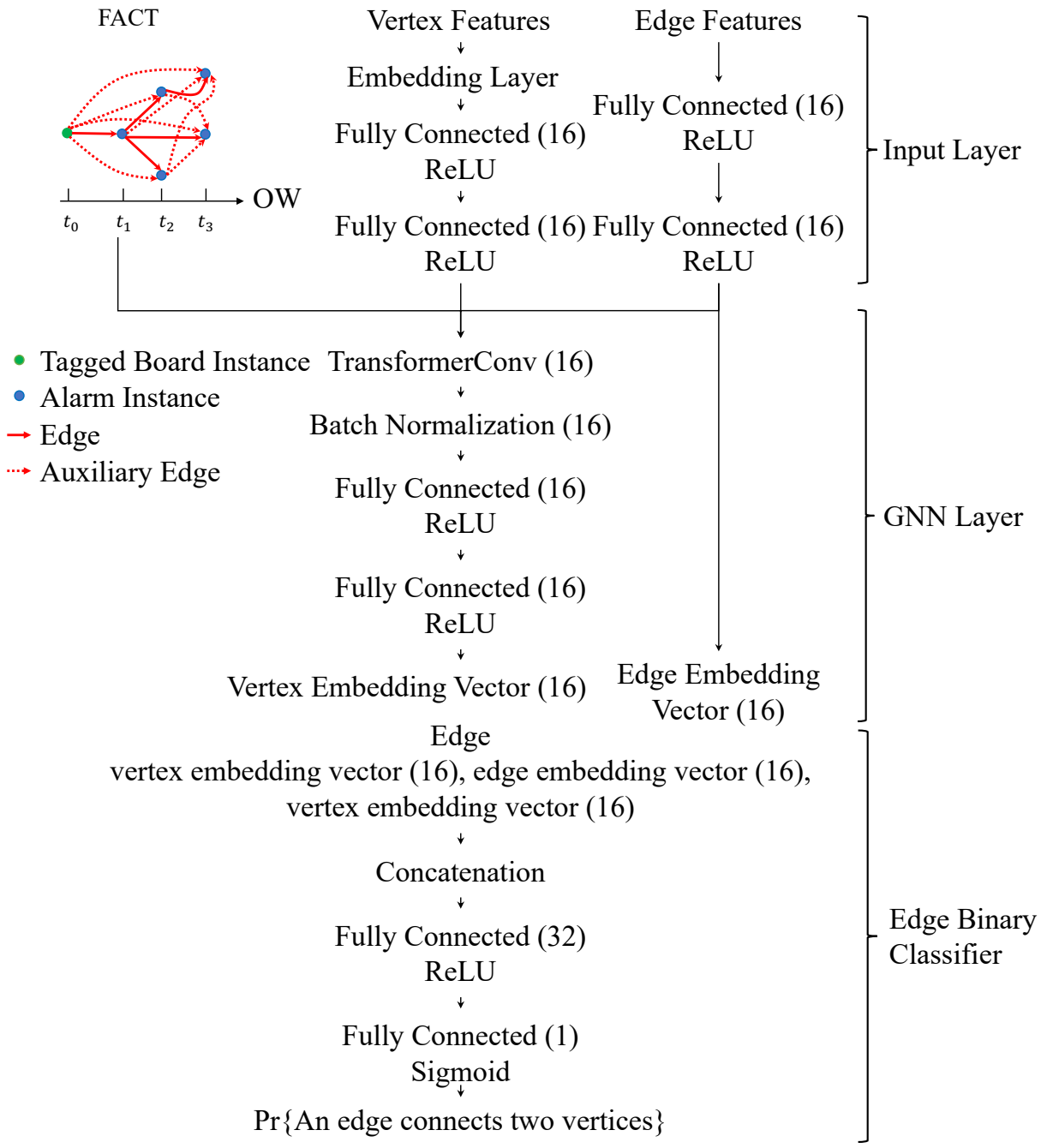


Figure 4.3: Architecture of the graph edge binary classifier.

4.2.3 FG Construction

Within each OW P, the testing dataset of a new observation is given, which embraces the network topology T_P, traffic distribution L_P, and alarm set A_P. The presented FG construction process is designed to build a set of FGs, providing the search space for the subsequent FACT formation. Initially, a device board from T_P is labeled as the tagged board, which is used to generate a subgraph induced by T_P. This subgraph contains all boards going through the tagged board's functional connections, i.e., OTS, OMS, and OCH in the setting of OTN. Then we focus on the alarms reported by the boards in the subgraph and examine whether the tagged board could be associated with the subset of these alarms. Here, FACTs in the raw dataset can be leveraged for mining the failure-alarm association rules between the tagged board's failure type and alarm type(s) in each FACT. If the tagged board and its alarm subset satisfy the corresponding association rule, this subgraph is deemed as a valid FG otherwise we skip this tagged board. The above procedure will be performed for each device board in T_P, resulting in a set of FGs, where the tagged board and its alarm subset are attached to each FG as well.

4.2.4 FACT Formation

Given the trained graph edge binary classifier and FGs, the proposed FACT formation algorithm aims to construct the minimum cost FACTs while covering all alarm instances in A_P, with each FACT rooted by a failed board and taking some alarms as the leaves.

Fig. 4.4 showcases the proposed FACT formation algorithm. For each FG, a potential FACT can be generated by pairing its tagged board with each alarm from the alarm subset and connecting all alarm instance pairs with a positive occurrence time gap. Then each potential FACT is fed into the trained graph edge binary classifier for weighting its edges. These weighted potential FACTs will be post-processed with two steps. We take the graph union of all weighted potential FACTs, followed by checking whether the resultant graph is a simple graph. If there exists more than one edge between any two vertices, we merely reserve one of the maximum-weight edges. Afterward, a weighted directed acyclic graph $\mathcal{G}_P$ is prepared as the search space of optimal FACT(s), denoted as $\mathcal{G}_P = (V_P, E_P, C(E_P))$. $V_P = B_P \bigcup A_P$ represents the vertex set aggregated by the board instance set B_P and alarm set A_P. $E_P = E_P^1 \bigcup E_P^2$ is the edge set, where $E_P^1 = B_P \times A_P$, $E_P^2 \subseteq A_P \times A_P$ are the edge sets formed by boards and alarms as well as alarm pairs, respectively. Also, $C(E_P)$ is the set of non-negative edge costs, where $\forall v_i, v_j \in V_P, (v_i, v_j) \in E_P, c_{ij} \in C(E_P)$

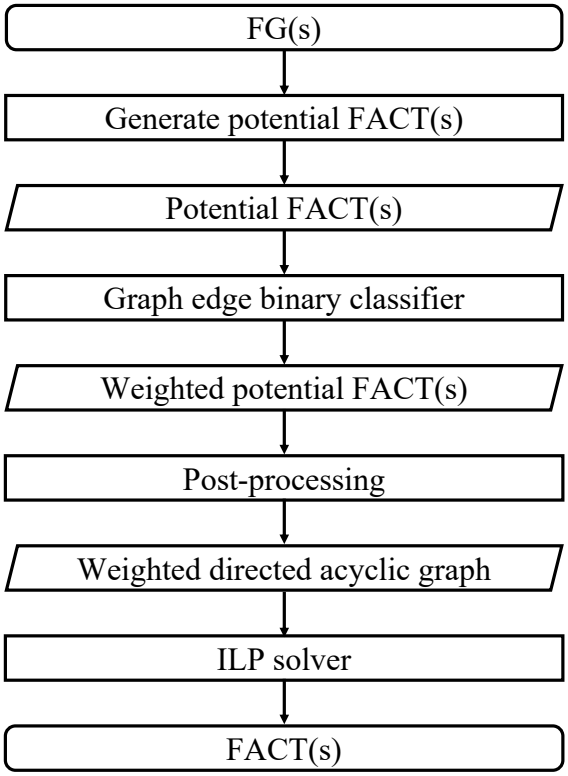


Figure 4.4: Flowchart of the proposed FACT formation algorithm.

signifies the weight/cost of edge (v_i, v_j) and it's defined in (4.1):

$$c_{ij} = \begin{cases} 1 - Pr\{v_i \text{ fails}\} \\ \quad \cdot Pr\{(v_i, v_j) \text{ connects } v_i, v_j\}, \text{if } (v_i, v_j) \in E_P^1, \\ 1 - Pr\{(v_i, v_j) \text{ connects } v_i, v_j\}, \text{if } (v_i, v_j) \in E_P^2, \end{cases} \tag{4.1}$$

where $Pr\{v_i \text{ fails}\}$ is the probability that board instance v_i fails. It can be derived from the probability density function (PDF) of the corresponding board's time-to-failure, where the PDF could be estimated grounded in historical failure events taking place at this board. $Pr\{(v_i, v_j) \text{ connects } v_i, v_j\}$ is the probability output from the trained graph edge binary classifier.

Furthermore, the problem of coming up with the optimal FACTs $\mathcal{G}_P^*$ can be framed as an ILP problem defined in (4.2):

$$\text{minimize} \quad \sum_{e \in E_P} c_e x_e + \sum_{u \in B_P} (1 - Pr\{u \text{ fails}\}) \cdot y_u \tag{4.2a}$$

$$\text{subject to} \quad \sum_{e \in \delta^-(v)} x_e = 1, \quad \forall v \in A_P, \tag{4.2b}$$

$$x_e \leq y_u, \quad \forall u \in B_P, \forall e \in \delta^+(u), \tag{4.2c}$$

$$x_e \in \{0, 1\}, \quad \forall e \in E_P, \tag{4.2d}$$

$$y_u \in \{0, 1\}, \quad \forall u \in B_P. \tag{4.2e}$$

$\forall e \in E_P, \forall u \in B_P$, two binary variables x_e, y_u are defined, where x_e takes 1 if the edge e is selected by $\mathcal{G}_P^*$ and 0 otherwise; while y_u takes 1 if the board instance vertex u is chosen as a tree root in $\mathcal{G}_P^*$ and 0 otherwise. The objective function (4.2a) aims to extract $\mathcal{G}_P^*$ that minimizes the total cost of opted edges and board instances. Constraint (4.2b) implies that for each alarm instance vertex v, only one item from the set of edges incoming to v, namely $\delta^-(v)$, must be chosen to ensure all alarm instances are visited by $\mathcal{G}_P^*$. Constraint (4.2c) suggests that for each board instance vertex u, if any item from the set of edges outgoing from u, called $\delta^+(u)$, is chosen, then u must be selected.

By solving the above ILP, the anticipated FACT(s) $\mathcal{G}_P^*$ is gained, which achieves both failure localization and alarm correlation.

4.3 Case Studies

Extensive case studies are performed to validate the viability of the proposed FACT-FL-GNN in the OTN optical layers and compare it with the counterparts. Firstly, we develop

an OTN simulator [48][44] to harvest ground-truth alarms and FACTs according to the given failure event, failure-alarm propagation rule database, as well as the network topology and traffic distribution during an OW. The rule database incorporates 38 entries in the form of *One2Many*, where 16 failure types, 16 board types, and 25 alarm types are taken into account. Without loss of generality, each failure event independently hits a certain number of boards and breaks their traversing functional connections. Besides, we assume that the failure-alarm propagation process always succeeds even though any failed board exists along the notification signal propagation path. On the other hand, we suppose that each board instance is equiprobable to fail, thereby ignoring the cost of choosing a board instance by setting $Pr\{u \text{ fails}\}$ as 1 in (4.2a).

The case studies aim to verify the generality and migratability of FACT-FL-GNN. For this purpose, we apply the graph edge binary classifier trained by the raw dataset to the testing datasets gathered from OTN environments with diverse failures, network topologies, traffic distributions, and noise alarms. The state-of-the-art counterparts for comparison include BP [50], LSTM [50], convolutional neural network (CNN) [77], BERT [35], GNN [46, 47, 45], GNN-Dual [34], FACT-FL-Heuristic [37][39], and FACT-FL-Chain [38].

4.3.1 Setup

Raw Dataset

The raw dataset is produced from a default network state. It comprises 8 nodes, 264 boards, and 20 lightpaths each going through 14 boards on average and its board-level average degree is 2.2. We collect 3086 alarms from this default network state via 264 1-minute OWs, where these OWs are initiated by the single-board failure events that in turn hit each board in the network topology.

Dataset

The raw dataset leads to the dataset of 264 FACTs for training the graph edge binary classifier. Note that the size of each FACT equals to one plus the number of alarms caused by the failed root board, which is co-determined by the invoked failure-alarm propagation rules, network topology, and traffic distribution. The dataset is split into training, validation, and test sets with a ratio of 7:2:1. We utilize the min-max scaling to normalize the numerical features. The binary cross entropy loss function is optimized with Adam at a

learning rate of 0.005. The batch size and number of epochs are set to 64 and 500, respectively. In addition, we monitor the value of the area under the curve on the validation set in each epoch and make use of early stopping [7] for mitigating overfitting.

Machine Learning Model Counterparts

The network architectures of machine learning model counterparts under the raw dataset are concisely delivered as follows. For BP and LSTM, two networks of dimension $120 \times 64 \times 32 \times 264$ and $120 \times 64 \times 264$ are established. The CNN model is made up of 25 input layer units and 3 hidden layers whose number of neurons are 256, 128, and 32, where the kernel size in each hidden layer is 3×3. As for BERT, it encodes the alarm contextual information with a 768-dimension vector and sets the length of each alarm transaction to 3. For training GNN, an alarm knowledge graph with 41 entities is structured based on the rule database. For GNN-Dual, one convolutional layer using the graph operator in [41] is utilized to obtain a 32-dimension feature vector for each node. Lastly, FACT-FL-Heuristic exerts a 3-layer fully connected neural network whose number of neurons is 32, 32, and 64, respectively.

Testing Datasets

The process of generating testing datasets is sketched out as follows. To assess the benchmark performance for each scheme, we emulate a single-board failure dataset of 10 failed boards randomly selected from the default network state.

Moreover, we generate two regional failure datasets to evaluate the generality and migratability of each method. One dataset is composed of regional failures from 10 new network states achieved by altering the number of lightpaths in a common network topology. This network topology consists of 10 nodes and 2380 boards with a board-level average degree of 2.06. The number of lightpaths ranges from 10 to 100 with an equal interval of 10 and each lightpath averagely traverses 14 boards. The other dataset contains the regional failures from 10 new network states each taking 40 lightpaths. These network states correspond to different topological sizes, where the number of nodes, number of boards, board-level average degree, and average number of boards traversed by each lightpath vary from 11 to 20, 461 to 700, 2.21 to 2.26, and 12 to 18, respectively. Note that each regional failure starts with selecting one board from the network topology at random, which is further expanded by arbitrarily adding one board at a time and ensuring that the chosen boards are connected. The number of failed boards in each regional failure changes from 1 to 10 and the average degree of each regional failure is 1.39.

In the end, the single-board failure dataset incorporating 10 failed boards is simulated to test FACT-FL-GNN's noise-resistant ability. For each failed board, different ratios of noise alarms are introduced on top of the true alarms.

ILP Solver and Performance Metrics

To deal with the ILP problem with a huge number of variables, we take the branch-and-price method [4], which is implemented in the high-performance optimization software (HiGHS) [23] that is dedicated to large-scale sparse linear optimization models.

Two performance metrics called *F-measure(F)* and *F-measure(R)* are considered for evaluating the two outcomes from our framework in each imitated instance, termed *failed boards* and *root alarms*. *F-measure(F)* and *F-measure(R)* are defined in (4.3) and (4.4), where a larger value implies better performance in identifying failed boards/root alarms:

$$\begin{aligned} F\text{-}measure(F) &= 2 \cdot \frac{Precision(F) \cdot Recall(F)}{Precision(F) + Recall(F)}, \\ Precision(F) &= \frac{\#\text{ of correctly identified failed boards}}{\text{total }\#\text{ of identified failed boards}}, \\ Recall(F) &= \frac{\#\text{ of correctly identified failed boards}}{\text{total }\#\text{ of failed boards}}, \end{aligned} \tag{4.3}$$

$$\begin{aligned} F\text{-}measure(R) &= 2 \cdot \frac{Precision(R) \cdot Recall(R)}{Precision(R) + Recall(R)}, \\ Precision(R) &= \frac{\#\text{ of correctly identified root alarms}}{\text{total }\#\text{ of inferred root alarms}}, \\ Recall(R) &= \frac{\#\text{ of correctly identified root alarms}}{\text{total }\#\text{ of root alarms}}. \end{aligned} \tag{4.4}$$

4.3.2 Results

Graph Edge Binary Classifier

The training curve of the graph edge binary classifier is shown in Fig. 4.5. The loss converges to 0.0862/0.0891 and the accuracy reaches 96.07%/95.75% after 463 epochs on the training/validation set.

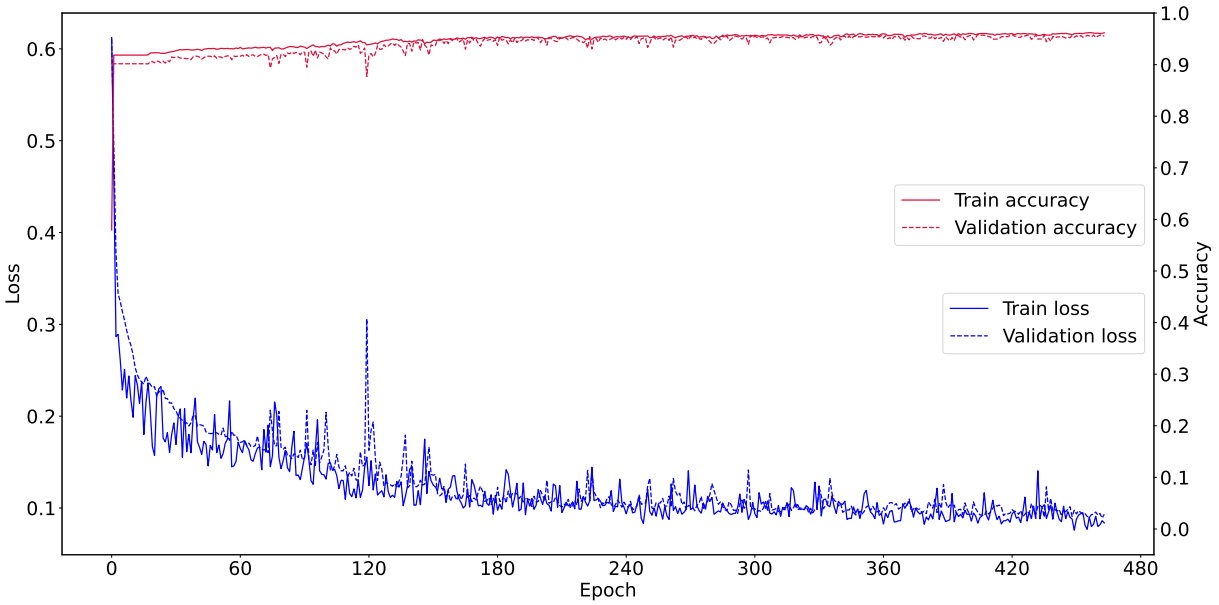


Figure 4.5: Training curve of the graph edge binary classifier.

FACT Formation

Table 4.2: Results on the single-board failure dataset for benchmark performance comparison

Method	# of trainable parameters	*F-measure(F)*	*F-measure(R)*
FACT-FL-GNN	4521	1.000±0.000	1.000±0.000
FACT-FL-Chain, $K = 3$	18771	0.9667±0.062	0.9±0.186
FACT-FL-Chain, $K = 2$	11490	0.9±0.186	0.8667±0.189
FACT-FL-Chain, $K = 1$	5265	0.9±0.186	0.7667±0.245
FACT-FL-Heuristic	5585	0.391±0.211	0.747±0.241
CNN	338378	N/A	0.630±0.272
BERT	2521646		0.200±0.211
BP	40904	0.500±0.310	N/A
LSTM	64520	0.500±0.310	
GNN	51999	0.800±0.248	
GNN-Dual	16801	0.2567±0.17	

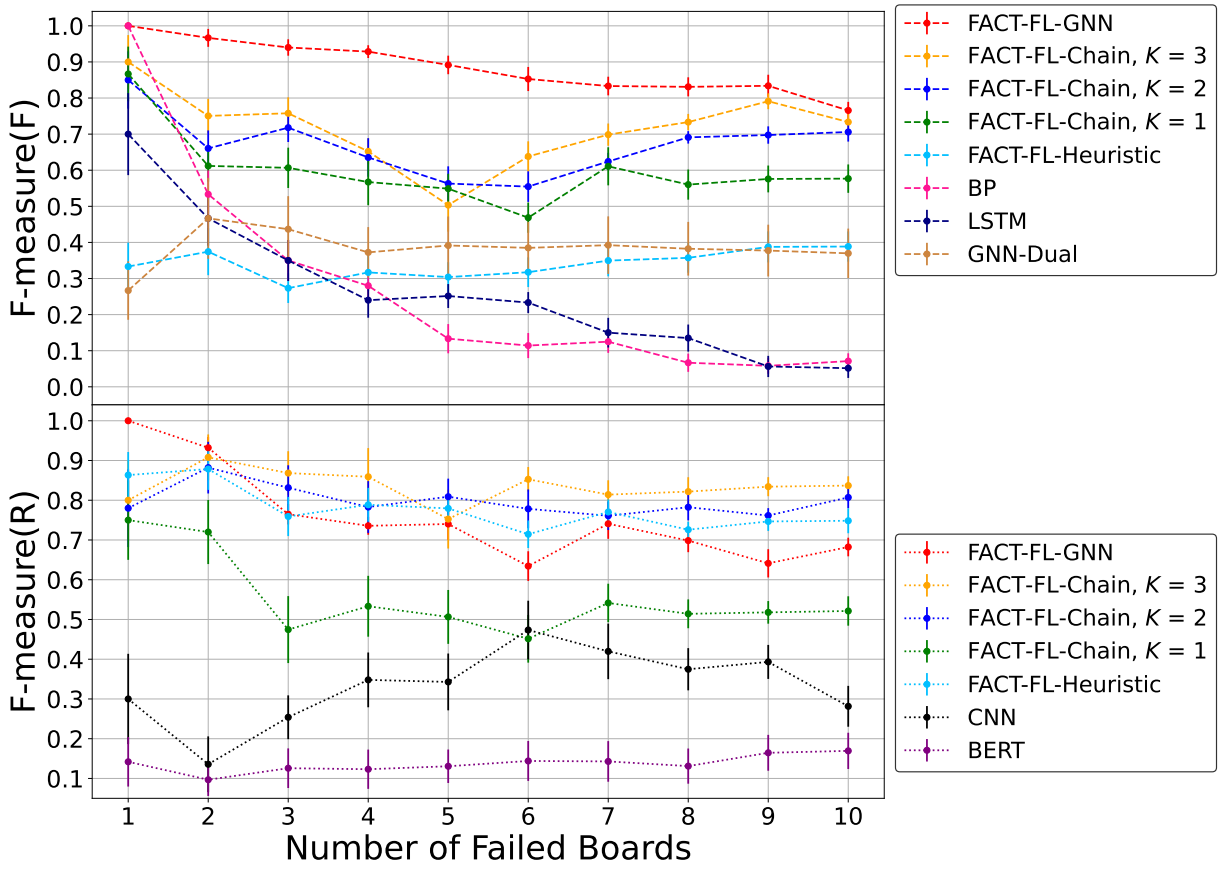


Figure 4.6: Average performance result on the regional failure dataset under various traffic distributions.

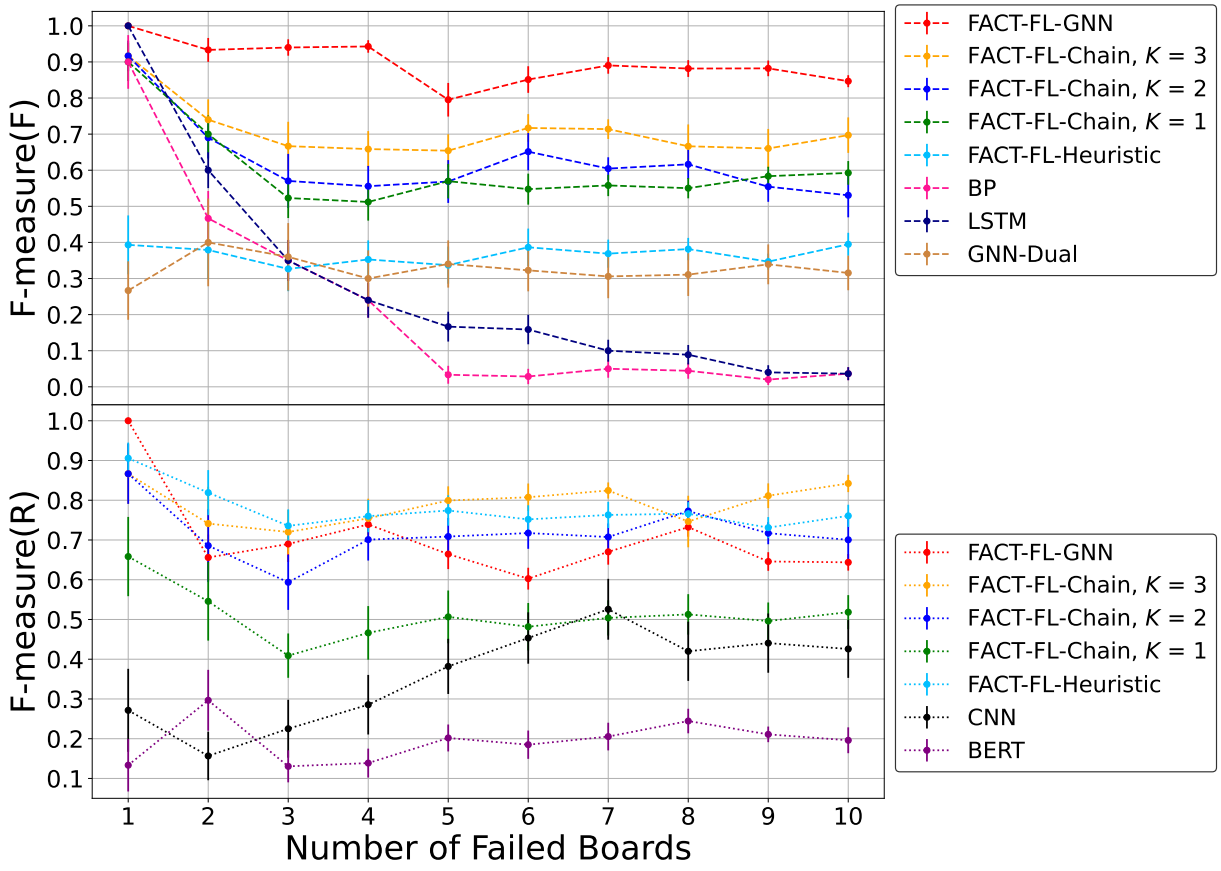


Figure 4.7: Average performance result on the regional failure dataset under various network topologies.

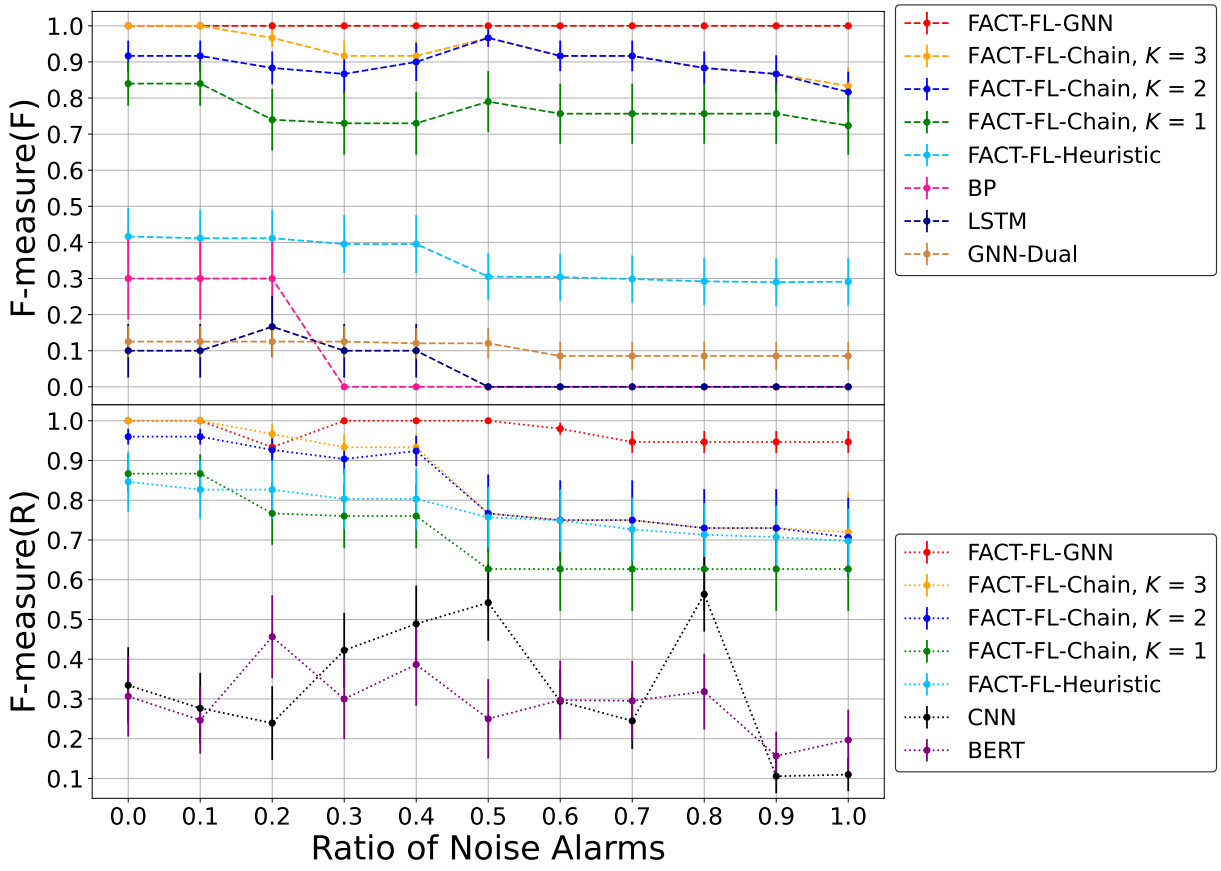


Figure 4.8: Average performance result on the single-board failure dataset under various ratios of noise alarms.

Firstly, the results of all schemes on the single-board failure dataset for benchmark performance comparison are listed in Table 4.2, where the value before "±" means the average *F-measure(F)/F-measure(R)* over 10 failure events and the value after "±" stands for the margin of error at the confidence interval of 95%. Notably, FACT-FL-GNN attains a pronounced advantage over its counterparts in all employed metrics while taking the fewest trainable parameters.

Furthermore, the results of all schemes on two regional failure datasets are summarized in Fig. 4.6 and Fig. 4.7. Obviously, FACT-FL-GNN performs the best among all its counterparts in recognizing failed boards. FACT-FL-Chain, which is second only to FACT-FL-GNN, behaves better with a larger value of K. Nevertheless, BP and LSTM grapple with a significant performance decline with increasing the number of failed boards, which hints that they could lose more real failed boards on account of neglecting abundant alarm attributes. Similarly, GNN-Dual, which is neck and neck with FACT-FL-Heuristic, is subject to poor performance due to overlooking the traffic distribution, even if the network topology is considered. Again, FACT-FL-GNN outperforms its counterparts in terms of root alarm identification except for FACT-FL-Chain with $K = 2$ or 3, and FACT-FL-Heuristic. However, CNN and BERT suffer from dramatic performance degradation because they completely ignore the traffic distribution and spatial relation among these alarms. Note that although FACT-FL-Chain and FACT-FL-Heuristic are on par with or even better than FACT-FL-GNN in recognizing root alarms, they are still worse than FACT-FL-GNN in localizing failed boards since they independently evaluate each edge weight without considering the hidden tree-shape correlation pattern between the tagged board and alarm subset.

Additionally, the results of various schemes on the single-board failure dataset with noise alarms are exhibited in Fig. 4.8. FACT-FL-GNN has the strong capability of noise immunity in identifying failed boards/root alarms thanks to its careful consideration of the tree-shape correlation among the received alarms. In contrast, most of its counterparts are severely degraded in detecting failed boards/root alarms with the increase in the ratio of noise alarms.

Further note that since the location information of failed boards/alarms changes through the evolving network environments, all models used by FACT-FL-GNN's counterparts (apart from FACT-FL-Heuristic and FACT-FL-Chain) require retraining when any disturbance occurs in the network topology or traffic distribution, but FACT-FL-GNN needs to be trained once with the raw dataset gathered from the default network state and thus it realizes the best migratability among all considered counterparts.

Complexity Analysis

We assume that during an OW P, the board set $B_P = \{b_1, \cdots, b_{M_P}\}$ and alarm set $A_P = \{a_1, \cdots, a_{N_P}\}$ are provided. The step of FG construction requires examining each board from B_P, leading to m_P FGs and the computation complexity of $\mathcal{O}(M_P)$. Furthermore, the complexity of the FACT formation algorithm derives from using the graph edge binary classifier and ILP solver. The former one results in $\mathcal{O}(m_P)$. The latter one is known as an NP-hard problem and thus it's overlooked in the complexity analysis, where the number of variables and constraints considered by the ILP in (4.2) encompass no more than $m_P + \frac{N_P \cdot (N_P - 1)}{2}$ and $N_P + m_P \cdot N_P$, respectively.

4.4 Conclusion

This chapter presents a novel approach for board-level failure localization in the OTN, called FACT-FL-GNN. The proposed FACT-FL-GNN is characterized by several impressive modeling approaches. Firstly, we have collaborated boards and alarms into the correlation process that pinpoints both failed boards and root alarms. Secondly, we have trained a graph edge binary classifier that is generalized to various network environments that deploy the same failure-alarm propagation rules. Thirdly, we have proposed the concept of FACT that effectively models the correlation among the failed boards and alarms, and the FACT formation procedure is formulated as a solvable ILP problem that provides a graceful solution.

Extensive case studies are conducted to compare the proposed FACT-FL-GNN with other schemes in diverse network environmental changes. We observe that FACT-FL-GNN has achieved extraordinary and stable performance in localizing failed boards and root alarms, which champions our claim that FACT-FL-GNN boasts sufficient generality and migratability in variable network environments.

Chapter 5

Conclusion

In this thesis, we propose a novel framework of board-level failure localization in the OTN optical layer, namely Failure-Alarm Correlation Tree based Failure Localization (FACT-FL), where its outcome is one or multiple FACT(s) each taking one failed board and its correlated alarms as the root and leaves, respectively. We have implemented three approaches to achieve FACT-FL. The proposed FACT-FL-Heuristic (in Chapter 2) designs a binary classifier that learns the correlation measure between a board instance and an alarm instance/two alarm instances, followed by raising a heuristic algorithm for creating the feasible FACT(s). Further, to improve FACT-FL-Heuristic's performance, we come up with FACT-FL-Chain (in Chapter 3), an approach that regards each FACT as a set of correlation chains with different order values and yields the appropriate FACT(s) by working out an ILP problem. Moreover, to reduce the computational complexity of generating all chain candidates for FACT-FL-Chain, we put forward FACT-FL-GNN (in Chapter 4), a method that adopts GNN for calculating the edge weights of latent FACT(s) and formulates a simplified ILP to obtain the most likely FACT(s). The results of extensive case studies have demonstrated that the proposed approaches' advantages over their counterparts in terms of the metrics evaluating the identified failed boards/root alarms, even if the network environment undergoes versatile variations including diverse failure scenarios, network topologies, traffic distributions, and noise alarms.

5.1 Contributions

The contributions of this thesis are summarized as follows:

- We have investigated a novel failure localization framework, called FACT-FL, which leans on the concept of FACT that effectively models the correlation among the failed board and alarms so as to achieve both failed board localization and root alarm identification.
- We have implemented three approaches to realize FACT-FL, namely FACT-FL-Heuristic, FACT-FL-Chain, and FACT-FL-GNN.
- We have developed various types of binary classifiers that incorporate all those static and dynamic network parameters, aiming at fulfilling sufficient generality and migratability for volatile network environments.
- We have framed the FACT formation procedure as a solvable ILP problem that offers graceful results.
- We have carried out abundant case studies to unveil that the presented approaches have achieved superb and stable performance in identifying failed boards and root alarms, which advocates our claim that the proposed approaches boast sufficient generality and migratability in variable network environments.

5.2 Future Work

The following issues could be further investigated in future work:

- The proposed framework currently assumes that the failure-alarm propagation rules are constant in various network environments. However, those rules could vary with the network environments, which brings about the problem of how to fine-tune the trained binary classifiers.
- The case studies are conducted on the synthetic dataset instead of the real-world dataset. Therefore, we need to collect some real datasets from OTN and verify the effectiveness of the proposed approaches.
- The assumption made by our approach is that the failure-alarm propagation behavior in OTN follows *One2Many*, whereas there could be more complicated forms such as *Many2One* or appending some triggering conditions.

- It may be difficult for the engineers who aren't deeply versed in statistical modeling to provide accurate reference FACTs as the training dataset, leading to introducing some false FACTs/alarms. Accordingly, an effective pre-processing procedure is required to create a more reliable training dataset.

- The raised framework is merely compared with other machine learning (ML) approaches, though, it should be further compared against traditional failure localization methods.

- The proposed framework may be subject to high computational cost in the setting of large-scale network topology or alarm flood and thus it can't realize failure localization in real time. Hence, we are anticipated to propose another novel approach that inspects the status of each board in real time and achieves equivalent or even better performance than the state of the arts in the dynamically changing network environment.

- The objective function of formulated ILP problems simply considers the linear combination of the likelihood of opted instance correlation and selected board instance, which could be further improved by making the weight adapt to the change of network environment.

- The constraints of considered ILP problems only meet the requirement of establishing one or a set of trees, ignoring the fact that the structure of each feasible FACT is associated with the board connectivity of the lightpaths traversed by the tree root board. Thus, we need to reformulate the ILP problems.

References

[1] Lisa Abele, Maja Anic, Tim Gutmann, Jens Folmer, Martin Kleinsteuber, and Birgit Vogel-Heuser. Combining knowledge modeling and machine learning for alarm root cause analysis. *IFAC Proceedings Volumes*, 46(9):1843–1848, 2013.

[2] Xiang Ao, Haoran Shi, Jin Wang, Luo Zuo, Hongwei Li, and Qing He. Large-scale frequent episode mining from complex event sequences with hierarchies. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(4):1–26, 2019.

[3] Jatin Babbar, Ahmed Triki, Reda Ayassi, and Maxime Laye. Machine learning models for alarm classification and failure localization in optical transport networks. *Journal of Optical Communications and Networking*, 14(8):621–628, 2022.

[4] Cynthia Barnhart, Ellis L Johnson, George L Nemhauser, Martin WP Savelsbergh, and Pamela H Vance. Branch-and-price: Column generation for solving huge integer programs. *Operations research*, 46(3):316–329, 1998.

[5] Sima Barzegar, Emanuele Virgillito, Marc Ruiz, Alessio Ferrari, Antonio Napoli, Vittorio Curri, and Luis Velasco. Soft-failure localization and device working parameters estimation in disaggregated scenarios. In *Optical Fiber Communication Conference*, pages Th1F–2. Optica Publishing Group, 2020.

[6] Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, et al. Interaction networks for learning about objects, relations and physics. *Advances in neural information processing systems*, 29, 2016.

[7] Yoshua Bengio. Practical recommendations for gradient-based training of deep architectures. In *Neural Networks: Tricks of the Trade: Second Edition*, pages 437–478. Springer, 2012.

[8] Ruichu Cai, Siyu Wu, Jie Qiao, Zhifeng Hao, Keli Zhang, and Xi Zhang. Thps: Topological hawkes processes for learning causal structure on event sequences. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[9] Ciena. Experts guide to optical transport networks. https://media.ciena.com/documents/Experts_Guide_to_OTN_ebook-Utilities-Edition.pdf, 2015. [Accessed 31-Aug-2022].

[10] Randall Davis, Howard E Shrobe, Walter Hamscher, Kären Wieckert, Mark Shirley, and Steve Polit. Diagnosis based on description of structure and function. In *AAAI*, volume 82, pages 137–142, 1982.

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[12] Philippe Fournier-Viger, Ganghuan He, Min Zhou, Mourad Nouioua, and Jiahong Liu. Discovering alarm correlation rules for network fault management. In *International Conference on Service-Oriented Computing*, pages 228–239. Springer, 2020.

[13] Alex Fout, Jonathon Byrd, Basir Shariat, and Asa Ben-Hur. Protein interface prediction using graph convolutional networks. *Advances in neural information processing systems*, 30, 2017.

[14] Ruoxuan Gao, Lei Liu, Xiaomin Liu, Huazhi Lun, Lilin Yi, Weisheng Hu, and Qunbi Zhuge. An overview of ml-based applications for next generation optical networks. *Science China Information Sciences*, 63:1–16, 2020.

[15] Rentao Gu, Zeyuan Yang, and Yuefeng Ji. Machine learning for intelligent optical networks: A comprehensive survey. *Journal of Network and Computer Applications*, 157:102576, 2020.

[16] Takuo Hamaguchi, Hidekazu Oiwa, Masashi Shimbo, and Yuji Matsumoto. Knowledge transfer for out-of-knowledge-base entities: A graph neural network approach. *arXiv preprint arXiv:1706.05674*, 2017.

[17] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

[18] Jiawei Han, Jian Pei, Behzad Mortazavi-Asl, Helen Pinto, Qiming Chen, Umeshwar Dayal, and Meichun Hsu. Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth. In *proceedings of the 17th international conference on data engineering*, pages 215–224. IEEE, 2001.

[19] Kimmo Hatonen, Mika Klemettinen, Heikki Mannila, Pirjo Ronkainen, and Hannu Toivonen. Tasa: Telecommunication alarm sequence analyzer or how to enjoy faults in your network. In *Proceedings of NOMS'96-IEEE Network Operations and Management Symposium*, volume 2, pages 520–529. IEEE, 1996.

[20] Jiajin He and Hui Zhao. Fault diagnosis and location based on graph neural network in telecom networks. In *2020 International Conference on Networking and Network Applications (NaNA)*, pages 304–309. IEEE, 2020.

[21] Hristo Hristov. Graph attention networks — baeldung on computer science. https://www.baeldung.com/cs/graph-attention-networks, 2023. [Accessed 31-Aug-2023].

[22] J Huang, Z Chen, B Zhu, Y Jing, Y Liang, X Yang, R Li, J Han, and Y Zhao. Multi-fault localization based on knowledge graph combined with network topology in optical networks. In *Ninth Symposium on Novel Photoelectronic Detection Technology and Applications*, volume 12617, pages 1093–1098. SPIE, 2023.

[23] Qi Huangfu and JA Julian Hall. Parallelizing the dual revised simplex method. *Mathematical Programming Computation*, 10(1):119–142, 2018.

[24] Huawei. Optix osn 8800/6800/3800 v100r008c10 hardware description. https://support.huawei.com/enterprise/en/optical-transmission/optix-osn-8800-pid-16323?category=learn-about-products&subcategory=hardware-description, 2014. [Accessed 13-Mar-2023].

[25] Huawei. Optix osn 8800/6800/3800 v100r009c10 alarms and performance events reference. https://support.huawei.com/enterprise/en/transmission-network/optix-osn-8800-pid-16323?category=troubleshooting, 2017. [Accessed 2-Sept-2022].

[26] Huawei. Wdm site types. https://info.support.huawei.com/network/ptmngsys/Web/WDMkg/en/24_wavesite.html, 2017. [Accessed 1-Sept-2022].

[27] Huawei. What is the wdm trail? https://forum.huawei.com/enterprise/en/what-is-the-wdm-trail/thread/835869-875, 2017. [Accessed 2-Sept-2022].

[28] Huawei. Osn 8800 6800 3800 v100r009c10 hardware description 03. https://support.huawei.com/enterprise/en/doc/EDOC1000079083, 2020. [Accessed 28-Aug-2023].

[29] ITU-T. G.709: Interfaces for the optical transport network. https://www.itu.int/rec/T-REC-G.709, 2020. [Accessed 28-Aug-2022].

[30] ITU-T. Optical transport network (otn) tutorial. https://www.itu.int/ITU-T/studygroups/com15/otn/OTNtutorial.pdf, 2020. [Accessed 28-Aug-2023].

[31] ITU-T. Classification on imbalanced data:tensorflow core. https://www.tensorflow.org/tutorials/structured_data/imbalanced_data., 2023. [Accessed 4-Aug-2023].

[32] Gabriel Jakobson and Mark Weissman. Alarm correlation. *IEEE network*, 7(6):52–59, 1993.

[33] Gabriel Jakobson and Mark Weissman. Real-time telecommunication network management: Extending event correlation with temporal constraints. In *International Symposium on Integrated Network Management*, pages 290–301. Springer, 1995.

[34] Jinglong Ji, Fujin Zhu, Jiaxu Cui, Haihong Zhao, and Bo Yang. A dual-system method for intelligent fault localization in communication networks. In *ICC 2022-IEEE International Conference on Communications*, pages 4062–4067. IEEE, 2022.

[35] Jinwei Jia, Danshi Wang, Chunyu Zhang, Hui Yang, Luyao Guan, Xue Chen, and Min Zhang. Transformer-based alarm context-vectorization representation for reliable alarm root cause identification in optical networks. In *2021 European Conference on Optical Communication (ECOC)*, pages 1–4. IEEE, 2021.

[36] Wenhao Jiang and Yuebin Bai. Apgnn: Alarm propagation graph neural network for fault detection and alarm root cause analysis. *Computer Networks*, 220:109485, 2023.

[37] Yan Jiao, Pin-Han Ho, Xiangzhu Lu, Kairan Liang, Yuren You, János Tapolcai, Bingbing Li, and Limei Peng. On real-time failure localization via instance correlation in optical transport networks. In *2023 IFIP Networking Conference (IFIP Networking)*, pages 1–9. IEEE, 2023.

[38] Yan Jiao, Pin-Han Ho, Xiangzhu Lu, Janos Tapolcai, and Limei Peng. A novel framework for optical layer device board failure localization in optical transport network. *IEEE Transactions on Network and Service Management*, 2023.

[39] Yan Jiao, Pin-Han Ho, Xiangzhu Lu, Yuren You, Janos Tapolcai, and Limei Peng. A novel framework of failure localization in optical transport network. *IEEE Communications Magazine*, 2023.

[40] Elias Khalil, Hanjun Dai, Yuyu Zhang, Bistra Dilkina, and Le Song. Learning combinatorial optimization algorithms over graphs. *Advances in neural information processing systems*, 30, 2017.

[41] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[42] Mika Klemettinen, Heikki Mannila, and Hannu Toivonen. Rule discovery in telecommunication alarm data. *Journal of Network and Systems Management*, 7(4):395–423, 1999.

[43] John Boaz Lee, Ryan A Rossi, Sungchul Kim, Nesreen K Ahmed, and Eunyee Koh. Attention models in graphs: A survey. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 13(6):1–25, 2019.

[44] Zening Li, Pin-Han Ho, Yan Jiao, Bingbing Li, and Yuren You. Design of an otn-based failure/alarm propagation simulator. In *2022 International Conference on Networking and Network Applications (NaNA)*, pages 1–5. IEEE, 2022.

[45] Zhuotong Li, Yongli Zhao, Yajie Li, Sabidur Rahman, Feng Wang, Xiangjun Xin, and Jie Zhang. Fault localization based on knowledge graph in software-defined optical networks. *Journal of Lightwave Technology*, 39(13):4236–4246, 2021.

[46] Zhuotong Li, Yongli Zhao, Yajie Li, Sabidur Rahman, Ying Wang, Xiaosong Yu, Lizhong Zhang, Guoli Feng, and Jie Zhang. Demonstration of alarm knowledge graph construction for fault localization on onos-based sdon platform. In *Optical Fiber Communication Conference*, pages M3Z–14. Optica Publishing Group, 2020.

[47] Zhuotong Li, Yongli Zhao, Yajie Li, Sabidur Rahman, Xiaosong Yu, and Jie Zhang. Demonstration of fault localization in optical networks based on knowledge graph and graph neural network. In *2020 Optical Fiber Communications Conference and Exhibition (OFC)*, pages 1–3. IEEE, 2020.

[48] Li, Zening. Design of an otn-based failure/alarm propagation simulator. Master's thesis, University of Waterloo, 2022.

[49] Enlu Lin, Qiong Chen, and Xiaoming Qi. Deep reinforcement learning for imbalanced classification. *Applied Intelligence*, 50:2488–2502, 2020.

[50] Tianyang Liu, Haoyuan Mei, Qiang Sun, and Huachun Zhou. Application of neural network in fault location of optical transport network. *China Communications*, 16(10):214–225, 2019.

[51] Jian-Guang Lou, Qiang Fu, Yi Wang, and Jiang Li. Mining dependency in distributed systems through unstructured logs analysis. *ACM SIGOPS Operating Systems Review*, 44(1):91–96, 2010.

[52] Huazhi Lun, Xiaomin Liu, Meng Cai, MengFan Fu, Yiwen Wu, Lilin Yi, Weisheng Hu, and Qunbi Zhuge. Anomaly localization in optical transmissions based on receiver dsp and artificial neural network. In *2020 Optical Fiber Communications Conference and Exhibition (OFC)*, pages 1–3. IEEE, 2020.

[53] Heikki Mannila, Hannu Toivonen, and A Inkeri Verkamo. Discovery of frequent episodes in event sequences. *Data mining and knowledge discovery*, 1:259–289, 1997.

[54] Robert Mathonet, Herwig Van Cotthem, and Leon Vanryckeghem. Dantes: An expert system for real-time network troubleshooting. In *Proceedings of the 10th international joint conference on Artificial intelligence-Volume 1*, pages 527–530, 1987.

[55] Kayol S Mayer, Jonathan A Soares, Rossano P Pinto, Christian E Rothenberg, Dalton S Arantes, and Darli AA Mello. Soft failure localization using machine learning with sdn-based network-wide telemetry. In *2020 European Conference on Optical Communications (ECOC)*, pages 1–4. IEEE, 2020.

[56] Kayol S Mayer, Jonathan A Soares, Rossano P Pinto, Christian E Rothenberg, Dalton S Arantes, and Darli AA Mello. Machine-learning-based soft-failure localization with partial software-defined networking telemetry. *Journal of Optical Communications and Networking*, 13(10):E122–E131, 2021.

[57] Fiber Optical Networking. The otn online tutorial. https://www.fiber-optical-networking.com/the-otn-online-tutorial.html, 2018. [Accessed 1-Sept-2022].

[58] Emanuele Rossi, Bertrand Charpentier, Francesco Di Giovanni, Fabrizio Frasca, Stephan Günnemann, and Michael Bronstein. Edge directionality improves learning on heterophilic graphs. *arXiv preprint arXiv:2305.10498*, 2023.

[59] Isabelle Rouvellou and George W Hart. Automatic alarm correlation for fault identification. In *Proceedings of INFOCOM'95*, volume 2, pages 553–561. IEEE, 1995.

[60] Alvaro Sanchez-Gonzalez, Nicolas Heess, Jost Tobias Springenberg, Josh Merel, Martin Riedmiller, Raia Hadsell, and Peter Battaglia. Graph networks as learnable physics engines for inference and control. In *International Conference on Machine Learning*, pages 4470–4479. PMLR, 2018.

[61] Yunsheng Shi, Zhengjie Huang, Shikun Feng, Hui Zhong, Wenjin Wang, and Yu Sun. Masked label prediction: Unified message passing model for semi-supervised classification. *arXiv preprint arXiv:2009.03509*, 2020.

[62] David I Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE signal processing magazine*, 30(3):83–98, 2013.

[63] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[64] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, Yoshua Bengio, et al. Graph attention networks. *stat*, 1050(20):10–48550, 2017.

[65] VIAVI. White paper g.709 – the optical transport network (otn). https://www.viavisolutions.com/en-us/literature/g709-optical-transport-network-otn-white-papers-books-en.pdf, 2021. [Accessed 1-Sept-2022].

[66] Danshi Wang, Liqi Lou, Min Zhang, Anthony C Boucouvalas, Chunyu Zhang, and Xuetian Huang. Dealing with alarms in optical networks using an intelligent system. *IEEE Access*, 7:97760–97770, 2019.

[67] Danshi Wang, Dongdong Wang, Chunyu Zhang, Lingling Wang, Songlin Liu, and Min Zhang. Machine learning for optical layer failure management. In *Optoelectronics and Communications Conference*, pages T3A–3. Optical Society of America, 2021.

[68] Danshi Wang, Chunyu Zhang, Wenbin Chen, Hui Yang, Min Zhang, and Alan Pak Tao Lau. A review of machine learning-based failure management in optical networks. *Science China Information Sciences*, 65(11):211302, 2022.

[69] Jiantao Wang, Caifeng He, Yijun Liu, Guangjian Tian, Ivy Peng, Jia Xing, Xiangbing Ruan, Haoran Xie, and Fu Lee Wang. Efficient alarm behavior analytics for telecom networks. *Information Sciences*, 402:1–14, 2017.

[70] Ruikun Wang, Jiawei Zhang, Shuangyi Yan, Chuidian Zeng, Hao Yu, Zhiqun Gu, Bojun Zhang, Tarik Taleb, and Yuefeng Ji. Suspect fault screening assisted graph aggregation network for intra-/inter-node failure localization in roadm-based optical networks. In *European Conference and Exhibition on Optical Communication*, pages We4B–4. Optica Publishing Group, 2022.

[71] Ruikun Wang, Jiawei Zhang, Shuangyi Yan, Chuidian Zeng, Hao Yu, Zhiqun Gu, Bojun Zhang, Tarik Taleb, and Yuefeng Ji. Suspect fault screen assisted graph aggregation network for intra-/inter-node failure localization in roadm-based optical networks. *Journal of Optical Communications and Networking*, 15(7):C88–C99, 2023.

[72] Yuying Wang, Huixuan Chi, and Qinfen Hao. Improving the robustness of graphsaint via stability training. *ParadigmPlus*, 2(3):1–13, 2021.

[73] Yongji Wu, Defu Lian, Yiheng Xu, Le Wu, and Enhong Chen. Graph convolutional networks with markov random field reasoning for social spammer detection. In *Proceedings of the AAAI conference on artificial intelligence*, pages 1054–1061, 2020.

[74] Xiaonan Xu, Haoshuo Chen, Jesse E Simsarian, Roland Ryf, Mikael Mazur, Lauren Dallachiesa, Nicolas K Fontaine, and David T Neilson. Optical network diagnostics using graph neural networks and natural language processing. In *Optical Fiber Communication Conference*, pages M3G–5. Optica Publishing Group, 2023.

[75] Hui Yang, Bohui Wang, Qiuyan Yao, Ao Yu, and Jie Zhang. Efficient hybrid multi-faults location based on hopfield neural network in 5g coexisting radio and optical wireless networks. *IEEE Transactions on Cognitive Communications and Networking*, 5(4):1218–1228, 2019.

[76] Hui Yang, Xudong Zhao, Qiuyan Yao, Ao Yu, Jie Zhang, and Yuefeng Ji. Accurate fault location using deep neural evolution network in cloud data center interconnection. *IEEE Transactions on Cloud Computing*, 10(2):1402–1412, 2020.

[77] Ao Yu, Hui Yang, Qiuyan Yao, Yajie Li, Huifeng Guo, Tao Peng, Haibin Li, and Jie Zhang. Accurate fault location using deep belief network for optical fronthaul networks in 5g and beyond. *IEEE Access*, 7:77932–77943, 2019.

[78] Chuidian Zeng, Jiawei Zhang, Ruikun Wang, Bojun Zhang, and Yuefeng Ji. Component fault location in optical networks based on attention mechanism with monitoring data. In *2022 European Conference on Optical Communication (ECOC)*, pages 1–4. IEEE, 2022.

[79] Chuidian Zeng, Jiawei Zhang, Ruikun Wang, Bojun Zhang, and Yuefeng Ji. Multiple attention mechanisms-driven component fault location in optical networks with network-wide monitoring data. *Journal of Optical Communications and Networking*, 15(7):C9–C19, 2023.

[80] Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor Prasanna. Graphsaint: Graph sampling based inductive learning method. *arXiv preprint arXiv:1907.04931*, 2019.

[81] Keli Zhang, Marcus Kalander, Min Zhou, Xi Zhang, and Junjian Ye. An influence-based approach for root cause alarm discovery in telecom networks. In *Service-Oriented Computing–ICSOC 2020 Workshops: AIOps, CFTIC, STRAPS, AI-PA, AI-IOTS, and Satellite Events, Dubai, United Arab Emirates, December 14–17, 2020, Proceedings*, pages 124–136. Springer, 2021.

[82] Min Zhang and Danshi Wang. Machine learning based alarm analysis and failure forecast in optical networks. In *2019 24th OptoElectronics and Communications Conference (OECC) and 2019 International Conference on Photonics in Switching and Computing (PSC)*, pages 1–3. IEEE, 2019.

[83] Si Zhang, Hanghang Tong, Jiejun Xu, and Ross Maciejewski. Graph convolutional networks: a comprehensive review. *Computational Social Networks*, 6(1):1–23, 2019.

[84] Xudong Zhang, Yuebin Bai, Peng Feng, Weitao Wang, Shuai Liu, Wenhao Jiang, Junfang Zeng, and Rui Wang. Network alarm flood pattern mining algorithm based on multi-dimensional association. In *Proceedings of the 21st ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, pages 71–78, 2018.

[85] Xudong Zhao, Hui Yang, Huifeng Guo, Tao Peng, and Jie Zhang. Accurate fault location based on deep neural evolution network in optical networks for 5g and beyond. In *Optical Fiber Communication Conference*, pages M3J–5. Optical Society of America, 2019.

[86] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *AI open*, 1:57–81, 2020.